UNIVERSITY OF UDINE

Department of
Mathematics, Computer Science and Physics

CORSO DI DOTTORATO DI RICERCA IN:
INFORMATICA, SCIENZE MATEMATICHE E FISICHE

in convenzione con Fondazione Bruno Kessler

Ciclo XXXV

Semantics for vision-and-language understanding

CANDIDATE
Alex Falcon

SUPERVISOR
Prof. Oswald Lanz

CO-SUPERVISOR
Prof. Giuseppe Serra

XXXV — 2022

Institute Contacts
Dipartimento di Scienze Matematiche, Informatiche e Fisiche
Università degli Studi di Udine
Via delle Scienze, 206
33100 Udine — Italia
+39 0432 558400
https://www.dmif.uniud.it/

Author's Contacts
Via Aquileia, 2
33029 Villa Santina — Italia
+39 0432 558446
falcon.alex@spes.uniud.it

Ringraziamenti

Il più grande e sentito ringraziamento va ai miei genitori, Daniela e Francesco, e mio fratello Andrea, che mi hanno sempre supportato in questo lungo viaggio. Non di meno ringrazio i miei nonni, Antonietta ed Emilio, e tutti i miei cari: Riccardo, Federica, Michelle, Raffaella, Nadia, Claudio, Cristina, Sergio. Senza tutti voi questo percorso sarebbe stato ancor più arduo.

Un immenso grazie va ai miei supervisori, Oswald Lanz e Giuseppe Serra, per tutte le cose che ho appreso in questi 3 anni.

Impossibile non ringraziare anche gli amici e colleghi per tutto il supporto ed aiuto offerto, oltre alle chiacchierate ed ore spese assieme: Nico, Fabio, Beatrice, Simone, Alessandro. Senza di voi questo lungo percorso sarebbe stato alquanto più solitario e meno piacevole.

Mandi!

29 Novembre 2022, Villa Santina

Alex Falcon

Abstract

Recent advancements in Artificial Intelligence have led to several breakthroughs in many heterogeneous scientific fields, such as the prediction of protein structures or self-driving cars. These results are obtained by means of Machine Learning techniques, which make it possible to automatically learn from the available annotated examples a mathematical model capable of solving the task. One of its sub-fields, Deep Learning, brought further improvements by providing the possibility to also compute an informative and non-redundant representation for each example by means of the same learning process. To successfully solve the task under analysis, the model needs to overcome the generalization gap, meaning that it needs to work well both on the training data, and on examples which are drawn from the same distribution but are never observed at training time. Several heuristics are often used to overcome this gap, such as the introduction of inductive biases when modeling the data or the usage of regularization techniques; however, a popular way consists in collecting and annotating more examples hoping they can cover the cases which were not previously observed. In particular, recent state-of-the-art solutions use hundreds of millions or even billions of annotated examples, and the underlying trend seems to imply that the collection and annotation of more and more examples should be the prominent way to overcome the generalization gap. However, there are many fields, e.g. medical fields, in which it is difficult to collect such a large amount of examples, and producing high quality annotations is even more arduous and costly.

During my Ph.D. and in this thesis, I designed and proposed several solutions which address the generalization gap in three different domains by leveraging semantic aspects of the available data. In particular, the first part of the thesis includes techniques which create new annotations for the data under analysis: these include data augmentation techniques, which are used to compute variations of the annotations by means of semantics-preserving transformations, and transfer learning, which is used in the scope of this thesis to automatically generate textual descriptions for a set of images. In the second part of the thesis, this gap is reduced by customizing the training objective based on the semantics of the annotations. By means of these customizations, a problem is shifted from the commonly used single-task setting to a multi-task learning setting by designing an additional task, and then two variations of a standard loss function are proposed by introducing semantic knowledge into the training process.

Sommario

I recenti sviluppi in Intelligenza Artificiale hanno portato al raggiungimento di importanti passi in avanti in diversi campi scientifici, quali la predizione della struttura delle proteine e gli autoveicoli a guida autonoma. Questi risultati sono stati ottenuti tramite tecniche di Machine Learning, il quale permette di apprendere in autonomia un modello matematico in grado di risolvere un problema a partire da degli esempi annotati. Il Deep Learning, che è uno dei campi di ricerca appartenenti al Machine Learning, ha portato ad ulteriori miglioramenti in quanto permette di ottenere anche una rappresentazione informativa e non ridondante di ogni esempio sfruttando lo stesso processo di apprendimento automatico. Per risolvere con successo il problema analizzato, il modello appreso necessita di superare il divario della generalizzazione (detto *generalization gap*), ovvero necessita di effettuare delle buone predizioni sia sui dati di addestramento, sia su esempi che sono campionati dalla stessa distribuzione ma non osservati in fase di addestramento. Varie euristiche sono spesso utilizzate per superare questo divario, quali l'introduzione di bias induttivi durante la fase di modellazione dei dati o l'uso di tecniche di regolarizzazione; tuttavia, un modo comunemente usato consiste nel collezionare ed annotare un numero maggiore di esempi nella speranza che coprano le casistiche non osservate in precedenza. In particolare, le soluzioni usate nello stato dell'arte utilizzano centinaia di milioni o anche miliardi di esempi annotati: l'andamento sempre crescente nel numero dei dati utilizzati sembra implicare che tale processo di collezione ed annotazione sia il modo principale per superare il *generalization gap*. Tuttavia tale processo non è facilmente applicare in molti campi, quale ad esempio il settore medicale, in cui è difficile collezionare un numero così alto di esempi, ed è ancor più arduo e costoso fornire delle annotazioni di qualità.

Durante il mio Corso di Dottorato ed in questa tesi, ho progettato e proposto varie soluzioni che riducono tale *gap* in tre domini diversi sfruttando degli aspetti semantici presenti nei dati a disposizione. In particolare, la prima parte di questa tesi include diverse tecniche che creano nuove annotazioni per i dati analizzati: tali tecniche comprendono strategie, chiamate di *data augmentation*, che ottengono variazioni dei dati e delle loro annotazioni tramite trasformazioni che ne preservano la semantica, e tecniche di trasferimento dell'apprendimento, che in questa tesi sono state usate per generare automaticamente descrizioni testuali per un insieme di immagini. Nella seconda parte della tesi, il divario è ridotto grazie ad una personalizzazione delle funzioni obiettivo usate in fase di addestramento basata sulla semantica delle annotazioni. In questa tesi, tramite queste variazioni è stato possibile trasformare un problema dall'impostazione tipicamente singolare ad un'impostazione *multi-task*, in cui cioè il modello impara a risolvere più problemi in contemporanea; successivamente, a partire da una funzione obiettivo standard sono state ottenute due variazioni che introducono aspetti di seman-

tica direttamente nel processo di addestramento.

Contents

1 Introduction

In recent years, Artificial Intelligence (AI) has hit the headlines of several news outlets showcasing outstanding achievements in highly complex problems, such as the generation of realistic images following a natural language description[1], or the prediction of the structure of proteins[2]. Many of these results in AI were possible thanks to recent advancements made in *Machine Learning* (ML), which is a branch of AI, and in particular in *Deep Learning* (DL), a subfield of ML. Instead of defining every exact detail used to solve a given problem, ML focuses on developing algorithms which automatically learn from available examples how to do it. In 1959, Arthur Samuel used it for the game of checkers, making the computer able to "learn from experience", in order to "play a better game of checkers than can be played by the person who wrote the program" [241]. To learn in this way, ML algorithms need to have a numerical representation of the input data which is informative and non-redundant: for instance, images can be represented as 3-D arrays, however keeping all the pixels is redundant and also unlikely to be informative, e.g. two images which contain a lot of blue would be considered similar, although one could show an airplane in the sky whereas a boat in the middle of the ocean could be the focus of the second image. Therefore, only the most important characteristics of the input data should be considered: these characteristics, called *features*, are intended to be informative and non-redundant, and are derived from the input data via a *feature extraction* process. Popular algorithms used to manually extract features for visual data include the Scale-Invariant Feature Transform (SIFT) algorithm [179], which locates 'key points' of the image and produces for each of them a scale- and rotation-invariant local representation, and the Histogram of Oriented Gradients (HOG) [43], which computes a global representation by using the distribution of the direction of the gradients. Differently from these techniques, DL aims at automatizing the feature extraction step by learning at the same time how to extract useful features from the raw input data, and how to solve the target task. The surge of DL started in 2012, after the impressive improvements obtained in *image recognition* with very deep neural networks [146]. Since then several breakthroughs were made in many highly

[1] https://www.nytimes.com/2022/04/06/technology/openai-images-dall-e.html
[2] https://www.bbc.com/news/science-environment-57929095

heterogeneous domains, ranging from self-driving cars [16], to achieving superhuman proficiency in complex games [258], to protein structure prediction [128].

To obtain these results, the *model* learned by any of these techniques needs to grasp the important details of the data which apply both to seen and unseen examples: in fact, since both ML and DL learn from a finite set of examples, there may be cases which are not covered and therefore never observed by the model during its learning phase. This problem has important relations to the *generalization gap*, which is intuitively defined as the difference between the performance measured on the training data and on another set of examples drawn from the same distribution, and is quite important to understand whether a trained model will operate well outside the learning environment. The generalization ability of a model could be verified by using the true distribution of data and computing this value. However, the true distribution is commonly unknown. To mitigate this problem, a subset of the available data is commonly held out, never used to gain any information during the learning phase, and only used to quantify the generalization ability: in particular, the available data is usually split into three sets, namely *training*, *validation*, and *test* set, which are respectively used as learning examples, a set of examples which are used to analyze the generalization after each update of the model, and finally the set of unseen examples which are only tested once in order to measure how well the learned model generalizes. By using these sets, it is possible to track the learning, to identify failure modes, and to introduce heuristics for a better generalization ability of the model; however, collecting more training data is often preferred, since it may lead to examples which have similar characteristics to the unseen ones. A naive idea could be to manually collect and annotate more examples: although tedious and error-prone, it would be feasible for a handful of researchers or other experts to provide high quality annotations. Yet, the most important limitation of this idea is about the quantity: in fact, some of the datasets used to tackle the aforementioned tasks are made of hundreds of thousands or even millions of annotated examples. It is clear that manually annotating such an amount of data becomes rapidly unfeasible. A clever next step consists in enlarging the pool of annotators, for instance by leveraging online *crowdsourcing* platforms which distribute the execution of tasks requiring human supervision across the globe. These platforms, which for instance include Amazon Mechanical Turk[3] and Prolific Academic[4], were used to collect several public datasets over time. In the TGIF dataset, 100000 animated GIFs collected on Tumblr were annotated by crowdworkers with 120000 free-form descriptions [166]. A Microsoft Research team collected in a similar way a total of 200000 natural language descriptions for 10000 videos retrieved from YouTube [310]. Even richer and more complex annotations were collected in the Visual Genome, including 3.8 millions of object instances, 1.7 millions visual questions and answers, and 5.4 millions of sentences describing specific regions of an image [145]. More recently, more than 3000 hours of video were collected in Ego4D and annotated with millions of different annotations, including dense textual descriptions of fine-grained activities, visual queries, and speech transcriptions, requiring more than 250000 hours of efforts from human annotators [87]. However, although the collection and annotation of these large scale datasets became feasible thanks to crowdsourcing platforms, the tediousness of the tasks solved by the

[3] https://www.mturk.com/
[4] https://www.prolific.co/

human annotators, the need to implement clever methods to guarantee the quality of the annotations, and the costs required to pay the human annotators for the invaluable service they performed, may still represent an hindrance to the collection of even bigger datasets. The collection of billions of examples might seem to lead to diminishing returns in terms of added knowledge when compared to millions, yet these humongous amounts of data are used to train recent state-of-the-art approaches: in particular, this is the case for foundational models which often perform *multi-task learning*, that is they try to solve multiple tasks at the same time by learning representations which are general enough to solve them all. Notable examples of this family of models include Flamingo which used around 500 millions of video-text and image-text pairs [3]; Florence which used 900 millions of image-text pairs [322]; ALIGN and SimVLM which used around 1.8 billions of image and alt-text pairs [123, 291]. Although crowdsource-based approaches may still be used to collect such a sizable amount of data, the costs for the annotation may become far too high. Therefore, several techniques capable of collecting the annotations automatically were proposed to reduce the costs. For instance, the annotation for a video could be given by any textual description attached to it, such as the alt-text used for accessibility purposes [123, 12], or by obtaining the speech transcript as generated by Automatic Speech Recognition (ASR) techniques [195]. Clearly, while these annotations are collected at a much cheaper cost, their quality may be lackluster when compared to those obtained by human annotators, and may also introduce several errors, e.g. in the case of ASR.

While there is a need for large amounts of annotated data to train DL models able to solve these tasks, the underlying trend seems to imply that the only way to overcome the generalization gap is to collect and annotate more and more data. However, there are two important shortcomings to this approach. First of all, it is not always possible to perform this collection and annotation step with ease: for instance, there are many domains in which the data is difficult to collect and even more problematic to annotate, such as when dealing with a datum which requires extensive expertise to correctly understand what it means and what it entails, e.g. tumor scans obtained with Magnetic Resonance Imaging. Secondly, while having more data is useful, there are many other ways to overcome the generalization gap, such as by working on the modeling aspects of the data or by paying more attention to the training procedure. During my Ph.D. studies I concentrated my efforts on these aspects, providing further evidence that the available data can be used to a greater extent in order to obtain better solutions which reduce the generalization gap. In particular, in this thesis such an objective is tackled by focusing on two main questions. In the first one, which covers Part I, the novel techniques we developed show how the available data can be used to generate new annotations which are helpful at training time. In the second one, covering Part II, the attention is moved to the learning aspects and in particular to the customization of the training objective, in which we propose to leverage semantic aspects of the available data to provide further or more target-oriented supervision, resulting in models which generalize better. A third technique to reduce the generalization gap consists in the development of new methods to model the data under analysis. Solutions of this type were explored during my studies, but on a highly different field of study, that of Predictive Maintenance. For this reason, Chapter 9 covers this topic and is part of the Appendix. Before moving to the two core parts of this thesis, the next section covers the

background which is useful to have an in-depth understanding of the following work. In particular, it presents three problems, which are Video Question Answering, Text-Video Retrieval, and Text-to-Image Synthesis, which were used as case studies to apply and test the proposed solutions. Then, an outline of the thesis will be presented, highlighting the main motivation and results, which also ended up in scientific products, as detailed in the list of publications presented thereafter.

1.1 Background

1.1.1 Video Question Answering

Video Question Answering, or VideoQA, requires to predict the correct answer to a natural language question about the visual contents of a video clip. Solving this problem has several interesting consequences, such as to provide support in training of human workers, to help and provide some visual context to visually impaired people, and also for more foundational questions related to the visual Turing test [83].

Following the general definition of the problem, multiple tasks stemmed from it based on the methodology used to deal with the answer reasoning and prediction phases. The closest approach to real-world scenarios, which is often termed as *open-ended*, *free-form*, or *generation-based*, requires the model to generate the answer word-by-word by solely reasoning on the question and the video [218, 306]. To simplify the answer prediction methodology, especially considering that the answers were often quite simple, classification-based objectives and metrics became the center of the attention in the community. The two most prominent approaches are called *multiple choice* and *open-ended*; in the former, a small pool of candidate answers (e.g. five choices in [72, 120, 273]) is provided along the video and the question, and the model needs to understand which candidate is correct; in the latter, the answer needs to be selected from a fixed set of possible answers from the dataset, e.g. 1000 in [308, 321, 327]. Since in the multiple choice task the model only needs to determine the correct answer among the candidates instead of answering the question, it is considered less challenging than the open-ended tasks [129]; therefore, recent works are trying to raise more attention towards the open-ended and especially the generation-based version, which are more of interest for real-world applications [306]. Another approach, called *fill-in-the-blank*, which consists in asking the model to fill the blanks in a template was also considered in some early works [186, 348], but it is getting more attention recently [23, 238].

Considering that VideoQA requires a mixture of computer vision and natural language processing, most of the methods developed for this task inherit several ideas from unimodal methodologies, customizing them to model the cross-modal interactions needed to solve the problem. About the textual data, it is common to extract pre-trained vectors for the word-level representations, such as by using Word2Vec [196] or GloVe [214], followed by a recurrent network to produce a context [171, 250]; some works also obtain a sentence-level representation directly, e.g. by using RoBERTa [189]. To compute a representation of the video, deep neural networks are used to capture regional [233], appearance [98, 272], and motion features [94, 276]; audio features may also play a role in identifying the correct answer to a question and therefore are used, especially within datasets containing audio-related questions [316, 323]. After obtaining

a representation for both types of data, attention mechanisms are often used to filter out irrelevant information and to attend solely to the most important frames or regions within the visual data, eventually guided by the question context [120, 140, 154, 163]. When regional features are added into the process, the interactions between frame- and region-level features are modelled with graphs [111, 250], possibly obtaining richer interactions by leveraging the textual context [48, 124, 171, 210]. A joint representation of video and question is hence computed and used to predict the correct answer. This step may be performed as simply as by using a classifier or a regressor, but it is common to model further cross-modal interactions in a late fusion approach. While attention mechanisms may still play a part into this process, more complex techniques were developed, involving multi-step reasoning techniques [73, 189] and the usage of memory layers to store and eventually make multiple modalities interact with each other [73, 79, 171]. Finally, similarly to what happened in other research fields, Transformers [279] also became popular in VideoQA [151, 157, 315]. However, their need for large scale pretraining datasets often leads to noisy data collected automatically from the web [195, 315].

1.1.2 Text-To-Video Retrieval

When looking for a video through a multimedia search engine, the user typically needs to provide a textual description of the expected contents; then, the engine outputs an ordered list of the videos contained in its database, in which the order is determined by how well a video is described by the query. By improving the solutions for this fundamental problem, it is possible to improve the multimedia search engines which are becoming more and more important as the amount of visual contents published online increases[5], but also to easily find photos or videos in private media galleries by providing a simple textual query. Moreover, given its multimodal nature, it has important relations to the general understanding of the cross-modal interactions happening between the visual contents and how we are used to describe them.

State-of-the-art solutions for the Text-To-Video Retrieval[6] problem are typically obtained by using deep learning techniques applied on large scale datasets, comprising at least one textual description (or caption) for each video clip. In particular, the most prominent approach consists in learning a joint text-video embedding space [12, 57, 184, 257], in which the representation of the video clip is forced to be highly similar to that of its own caption. By doing so, it is possible to search for a video by mapping the textual query into the same embedding space and then by ranking its neighbors through a similarity metric, such as the cosine similarity or the Euclidean distance. To achieve this goal, the loss function used to guide the optimization process is of utmost importance: considering the aforementioned peculiarities of the desired embedding space, a family of loss functions, called *contrastive*, is the most common choice since they aim at maximizing the similarity of the representations of video-caption pairs, while decreasing that of other pairs [36, 93, 193, 247]. From an architectural point of view, many heterogeneous approaches were proposed by the community. Given the multimodal nature of the problem and the availability of models (or "experts") pretrained

[5] Users of YouTube uploaded more than 500 hours of video every minute, as of February 2020 [25], and 95 millions of video and images are uploaded on Instagram daily [190].

[6] Commonly, Text-To-Video Retrieval is seen as a component of a more general setting, termed Text-Video or Video-Text Retrieval, which also considers the Video-To-Text aspect for a holistic evaluation.

for several tasks, it is common to compute a representation by mixing the information obtained by these multiple sources and by learning how to weigh each of the expert features. Notable examples include Mixture of Embedding Experts [194] and T2Vlad [288], which use NetVLAD [6] to perform the aggregation and matching of local features, Collaborative Experts [175], which introduce a collaborative mechanism to modulate the influence of each pretrained model according to the other experts, and TeachText [40] which extended the idea of using pretrained models to the language modeling aspect. Later works also used multimodal Transformers to model the cross-modal interactions through multiple self-attention layers [76, 61]. Differently from them, several researchers focused on leveraging the structure of the data coming from both modalities to improve the organization of the embedding space. By using Part-of-Speech (PoS) taggers to extract nouns and verbs from the descriptions, both PoS-restricted and PoS-agnostic embedding spaces were learned in [299]. The structure of the captions and the underlying semantic relations between its noun and verb phrases made it possible to learn hierarchical representations in [26], whereas later works extended this idea to also learn an hierarchical video graph [75, 127, 301] including sub-clip frame- or object-level representations. Recent works [52, 82, 212, 223, 236, 243] aim at leveraging additional linguistic supervision to inject semantic aspects into the visual representations, e.g. by using a support set of captions and a generative cost function [212], by obtaining additional language embeddings and pseudo-classnames to improve the semantic consistency of the visual representations [236], or by introducing a question-answering module which tries to solve a proxy task resembling the multiple choice setting of VideoQA [82].

Another aspect which is fundamental for Text-Video Retrieval and which is is observing a change in recent years consists in its evaluation. In fact, the evaluation was usually *instance-based*, that is given a query the evaluation metrics only consider where the "groundtruth" video is located in the ranking list: to do so, popular metrics include the recall rates, the median and mean rank, and the geometric mean of recall. Although these metrics represented the standard way to measure advancements in Text-Video Retrieval, they do not consider the quality of the ranking list, which is more indicative of high performance in retrieval systems: in fact, by using these metrics a model is penalized if, given a query, it ranks highly (i.e., far away from the top results in the ranking list) the video associated to that query in the dataset, despite ranking first a video which shares similar visual contents but a slightly different textual description, e.g. due to a synonym. Therefore, to truly quantify the advancements made in Text-Video Retrieval, the quality of the full ranking list would need to be assessed by means of more complex metrics, such as the nDCG [122] and the mAP [10]. In particular, these semantic nuances may influence highly on the performance, since multiple video clips may be described by similar captions. To use these metrics, the usage of semantic similarity measures was recently proposed to compute a proxy for the relevance grades, leading to *semantic* Text-Video Retrieval [45, 298].

1.1.3 Text-to-Image Synthesis

The generation of images conditioned on a natural language description takes the name of Text-to-Image Synthesis or Generation. Recent advancements on this topic, espe-

cially after the release of DALL-E 2 [224], hit several media outlets[7][8] and sparked an increased interest due to the high quality of the generated images. Apart from generating fancy images, being able to generate text-conditioned images affects several applications including image editing and graphic design.

With deep learning, this field of research rapidly improved and achieved interesting results. In particular, most of the state-of-the-art methods for Text-to-Image Synthesis are based on Generative Adversarial Networks (GAN) [85], which generally achieve a high sample quality [144, 168, 237, 330]. These networks follow an adversarial learning approach, in which two distinct networks, the "generator" and the "discriminator", compete to gain the upper hand. In particular, the "generator" generates increasingly more realistic images while trying to fool the "discriminator", which in turn learns to discriminate generated from real images. To have more control on the generation process, later works introduced a conditioning variable [200], especially focusing on a textual guidance [231]. Subsequent works started to use multiple GANs implementing a coarse-to-fine procedure to produce high resolution images [311, 331, 332], eventually supporting this procedure with additional structures, such as memories in which intermediate results and representations are stored [350], or additional training objectives, e.g. cycle consistency with an Image-to-Text objective [221, 347]. Given the success of contrastive loss functions in several cross-modal fields, several authors tried to steer the training paradigm towards these contrastive approaches [318, 319, 330]. In particular, after the release of the contrastively-trained text-vision model CLIP [223], many works also tried to integrate it into their methodologies [77, 78, 211]. For instance, Patashnik et al. proposed three different approaches for Text-to-Image manipulation which use CLIP to map a textual description of the desired change and identifying a direction in the latent space to perform such a transformation [211]; similarly, Gal et al. followed and adapted one of these approaches to enable out-of-domain Text-to-Image Synthesis [77].

However, the difficult training process and the inability to capture the high diversity of the true distribution represented important shortcomings of GANs which drove the attention to likelihood-based generative approaches [202, 227]. In particular, a class of likelihood-based models, called diffusion models [102, 261, 264], is getting increased attention since they have many desirable properties, e.g. ease of training and scalability, while also leading to high quality images [54, 102, 263, 264]. Many recent works are now investigating these models and their application to Text-to-Image Synthesis [8, 91, 126, 225, 235, 300], leading to a process which is typically made of two stages: in the first stage, an encoder-decoder architecture is learned by means of VQ-VAE [278] or VQ-GAN [64] to faithfully reconstruct an input image; then, based on a condition variable, the second stage either uses an autoregressive model to sequentially predict the image [225, 300], or a diffusion model to predict it by performing a gradual denoising process [91, 235].

[7] https://www.washingtonpost.com/technology/interactive/2022/artificial-intelligence-images-dall-e/

[8] https://www.nytimes.com/2022/04/06/technology/openai-images-dall-e.html

1.2 Thesis Outline and Contributions

The thesis is divided into two parts.

The first part, named "Can we reduce the generalization gap by automatically creating new labels?", aims at showing that the available annotations can be used to automatically create new ones, through generation or by using data augmentation techniques, and that the models resulting from the training with these additional annotations generalize better than the original ones. This initial part stems from the extensive work which was done over the years to mitigate the overfitting problem and to artificially increase the training dataset size. In this thesis, two techniques which are quite popular and widely used, especially in computer vision, became the center of the attention: data augmentation, and transfer learning. Data augmentation refers to the usage of semantics-preserving techniques which are used to create variations of the input data. These techniques became popular in computer vision after the work by Krizhevsky et al., who used random croppings, horizontal reflections, and changes in the RGB intensity to increase the training dataset size by a factor of 2048 [146]. Following this work, other researchers proposed other techniques for data augmentation, including the random erasing of image regions [341] or cutting a region from one image and applying it on another one [324]. Some of these techniques, especially the geometric or color-space transformations, were also applied to videos at the frame-level; nonetheless, the additional temporal dimension raises further possibilities to augment the input video clip, e.g. by temporally subsampling it [285] or by replacing part of it with a cuboid taken from a different video clip [325]. However, when textual annotations are paired to a video clip and used as part of the training data, these data augmentation techniques may introduce a discrepancy: for instance, the description of a video clip may refer to some object which is positioned on the "left"; when the video is horizontally flipped, its position on the image will change to the "right" and so the description needs to be updated to preserve the consistency and not to introduce errors in training. In Chapter 2, we detail our findings on this topic when applied to the Video Question Answering problem. We also design three data augmentation techniques which are applied to the raw textual and visual annotations, and experimentally validate them on a public dataset, obtaining an overall improvement of up to 5.5% and reducing the effect of several biases. Notwithstanding the popularity of these techniques, there are also a few disadvantages in their usage: the need for the data to be available and shareable, which may not always be possible since the videos may be taken from copyrighted contents, such as movies [273, 186] or TV series [153, 154], or may have been obtained from online platforms which might remove some of the previously published contents; the increased computational costs, because the full pipeline from raw pixels to target variables needs to be executed for each video; and the need to customize each data augmentation technique to the type of data under analysis, since for instance a technique designed for videos may not be usable for its textual description. A way to mitigate these problems consists in moving the augmentation to the latent space: in fact, by doing so the latent representations are the only elements which need to be shared and they do not raise copyright or privacy concerns; less computational resources are required, since forwarding the raw data across the full pipeline is no longer required; and, finally, the same techniques can be applied seamlessly to different modalities. Therefore, Chapter 3 presents a novel multimodal data augmentation technique which works in the latent

space: given a video and its own caption, their representations are mixed with those of another video-caption pair, which is identified within the dataset by quantifying the overlap of semantic concepts shared between the captions. We considered Text-Video Retrieval as a case study because it is commonly tackled by learning a joint embedding space, whose structure may benefit from learning the additional relationships occurring between different yet semantically similar videos and captions. When validated on on two public datasets, the proposed technique showed consistent improvements, highlighting its high robustness and effectiveness. The second technique considered in this part of the thesis, that is transfer learning, consists in learning how to solve a task on a big, often generic dataset in the hope that the knowledge learned there transfers well to other similar domains. When dealing with a narrow and niche domain, the public datasets for it may be designed for unimodal tasks, e.g. unconditioned image generation; however, cross-modal interactions are fundamental in order to have a control on the generation process, e.g. by means of a textual description. In our case study, described in Chapter 4, transfer learning was used to automatically generate the textual annotations for unlabelled visual data in order to learn Text-to-image Synthesis models. This was possible by learning a captioning model, and the experimental results confirmed the feasibility of this protocol.

In the second part of the thesis, named "Can we reduce the generalization gap by customizing the training objective?", the attention is driven to another important component of the learning pipeline: the training objective. The learning process which teaches a model how to solve a task through deep learning techniques involves the minimization of the cost as computed by a function, typically called loss function or objective function. At training time, a prediction is made for all the examples and an associated cost is computed, which is "high" for an error and "low" for a correct prediction; then, it is used to guide the learning process towards a minimum. Therefore, the choice of the right loss function, either among a set of standard ones or by designing it from scratch, is of utmost importance for the success of the overall learning process. In this thesis, particular attention was given to the customization of the training objective by using additional information which can be obtained from the available data. As a first case study, in Video Question Answering the learning process aims at minimizing the amount of times the wrong answer is chosen by the model in a set of possible candidate answers: to do so, a commonly used loss function computes a regression score for each of the candidates and aims at minimizing that of the wrong candidates, while maximizing the score of the correct answer. To reduce the overfitting, especially in domains with only few examples, and to improve the generalization ability of a model, an interesting possibility resides in the usage of multi-task learning: in fact, by requiring the model to solve multiple tasks at the same time, the internal representation computed by the model itself needs to be general enough and not specific for a single task. Therefore, in Chapter 5 we decided to introduce into the training objective the prediction of the type of the question. Although simple, such an objective was supported by empirical evidence which showed that the technique used to compute the representation for the textual annotations influences the performance obtained on specific question types. By using it, consistent improvements were achieved in an extensive benchmark covering multiple datasets, models, and word embedding techniques. Instead of adding customized functions into the training objective, in Chapters 6 and 7 the available data

is used to introduce semantic knowledge into standard loss functions. In particular, the case study under analysis is Text-Video Retrieval, which is commonly tackled by learning a joint embedding space in which the representation for a video is highly similar to that of its own captions: by doing so, a user formulates a query about the contents of the video in natural language, then the model maps this query into the same embedding space, which can then be used to find relevant videos by looking into the neighborhood of the query representation. To achieve this goal, a peculiar type of loss functions, called *contrastive*, is typically used because they aim at obtaining a similar representation for a video and its own caption, while decreasing the similarity with the other captions and videos [36, 247]. Although this methodology is widely used [257, 57], it implicitly assumes that only the captions provided for a video are able to describe its visual contents. However, this hardly holds in practice, since the same video may be described differently by two human annotators, for instance by using a highly diverse lexicon despite capturing the same visual aspects. To bring some of these semantic nuances into the training process, two different approaches are proposed. In Chapter 6, a fixed hyperparameter of a contrastive loss function is reformulated in terms of the overlap of semantic concepts computed between the captions associated to a video. By doing so, these semantic aspects are enforced directly onto the structure of the joint embedding space which is learnt at training time. By validating this methodology on three different network architectures and two datasets, we were able to confirm its effectiveness. Differently, in Chapter 7 the overlap of semantic concepts is used to separate the videos and captions into two different sets - namely, *relevant* and *irrelevant* to a given query example; then, the loss function is customized in order to implement different behaviors when dealing with examples from these two sets. The proposed strategy is validated across multiple models, datasets, and loss functions showing a high effectiveness and robustness.

Finally, Chapter 8 concludes this thesis by summarizing the main results and contributions, and by highlighting two research directions which may stem from them and which are filled with open questions and difficult challenges.

1.3 Publications

This Thesis is based on the following peer-reviewed works and publications:

1. Menardi, M., **Falcon, A.**, Mohamed, S. S., Seidenari, L., Serra, G., Bimbo, A. D., and Tasso, C. (2020). Text-to-Image Synthesis Based on Machine Generated Captions. In Proc. of Italian Research Conference on Digital Libraries (pp. 62-74). Springer, Cham.

   URL: https://doi.org/10.1007/978-3-030-39905-4_7

   This work sets the basis for Chapter 4.

2. **Falcon, A.**, Lanz, O., and Serra, G. (2020). Data augmentation techniques for the video question answering task. In Proc. of European Conference on Computer Vision Workshops (pp. 511-525). Springer, Cham.

   URL: https://doi.org/10.1007/978-3-030-66415-2_33

This work sets the basis for Chapter 2.

3. **Falcon, A.**, Serra, G., and Lanz, O. (2022). Learning video retrieval models with relevance-aware online mining. In Proc. of International Conference on Image Analysis and Processing (pp. 182-194). Springer, Cham.

   URL: https://dl.acm.org/doi/abs/10.1007/978-3-031-06433-3_16

4. **Falcon, A.**, Serra, G., and Lanz, O. (2022). Relevance-aware online mining for a more semantic video retrieval. In IEEE Transactions on Multimedia. *Under review.*

   Q1 in Computer Science Applications, Electrical and Electronic Engineering, Media Technology, and Signal Processing, **IF 8.182**

   These two works set the basis for Chapter 7.

5. **Falcon, A.**, Sudhakaran, S., Serra, G., Escalera, S., and Lanz, O. 2022. Relevance-based Margin for Contrastively-trained Video Retrieval Models. In Proc. of International Conference on Multimedia Retrieval (ICMR '22). Association for Computing Machinery, New York, NY, USA, 146–157.

   URL: https://doi.org/10.1145/3512527.3531395

   This work sets the basis for Chapter 6.

6. **Falcon, A.**, Serra, G., and Lanz, O. (2022). A Feature-space Multimodal Data Augmentation Technique for Text-video Retrieval. In Proc. of ACM International Conference on Multimedia (MM '22). Association for Computing Machinery, New York, NY, USA, 4385–4394.

   URL: https://dl.acm.org/doi/10.1145/3503161.3548365

   Conference rank: **A***

   This work sets the basis for Chapter 3.

7. **Falcon, A.**, Serra, G., and Lanz, O. (2022). Video question answering supported by a multi-task learning objective. In Multimedia Tools and Applications. *Under review.*

   Q1 in Media Technology, **Q2** in Computer Networks and Communications, Hardware and Architecture, and Software, **IF 2.577**

   This work sets the basis for Chapter 5.

Furthermore, during my Ph.D. and up to the thesis submission date (i.e., 2022-11-30), I produced the following peer-reviewed publications:

1. **Falcon, A.**, D'Agostino, G., Serra, G., Brajnik, G., and Tasso, C. (2020). A Neural Turing Machine-based approach to Remaining Useful Life Estimation. In Proc. of 2020 IEEE International Conference on Prognostics and Health Management (ICPHM) (pp. 1-8). IEEE.

URL: https://doi.org/10.1109/ICPHM49022.2020.9187043

2. **Falcon, A.**, D'Agostino, G., Serra, G., Brajnik, G., and Tasso, C. (2020, July). A Dual-Stream architecture based on Neural Turing Machine and Attention for the Remaining Useful Life Estimation problem. In Proc. of PHM Society European Conference (Vol. 5, No. 1, pp. 10-10).

   URL: https://doi.org/10.36001/phme.2020.v5i1.1227

3. **Falcon, A.**, D'Agostino, G., Lanz, O., Brajnik, G., Tasso, C., and Serra, G. (2022). Neural Turing Machines for the Remaining Useful Life estimation problem. Computers in Industry, 143, 103762.

   URL: https://doi.org/10.1016/j.compind.2022.103762

   Q1 in Computer Science (miscellaneous) and Engineering (miscellaneous), **IF 11.245**

4. D'Agostino, G., **Falcon, A.**, Lanz, O., Brajnik, G., Tasso, C., and Serra, G. (2022). Estimating the Remaining Useful Life via neural sequence models: a comparative study. To appear in Proc. of 2nd Italian Workshop on Artificial Intelligence and Applications for Business and Industries (AIABI), co-located with AI*IA 2022.

1.4 Other achievements and open source contributions

During my Ph.D., I also obtained the following awards:

Moreover, to support the reproducibility of the results, I publicly released on GitHub several open source codebases:

- *Neural Turing Machines for the Remaining Useful Life estimation problem*:

  https://github.com/aranciokov/NTM-For-RULEstimation

- *Relevance-aware online mining for a more semantic video retrieval* and *Learning video retrieval models with relevance-aware online mining*:

  https://github.com/aranciokov/ranp

- *A Feature-space Multimodal Data Augmentation Technique for Text-video Retrieval*:

  https://github.com/aranciokov/FSMMDA_VideoRetrieval

- *Relevance-based Margin for Contrastively-trained Video Retrieval Models*:

  https://github.com/aranciokov/RelevanceMargin-ICMR22

Can we reduce the generalization gap by automatically creating new labels?

2

Data augmentation techniques for the Video Question Answering task

2.1 Introduction

Video Question Answering (VideoQA) is a task that aims at building models capable of providing a meaningful and coherent answer to a visual contents-related question, exploiting both spatial and temporal information given by the video data. VideoQA is receiving attention from both the Computer Vision and the Natural Language Processing communities, due to the availability of both textual and visual data which require to be jointly attended to in order to give the correct answer [308, 79, 73].

Recent advancements in the VideoQA task have also been achieved thanks to the creation of several public datasets, such as TGIF-QA [120] and MSVD-QA [308], which focus on web scraped video that are often recorded from a third-person perspective. Even more recently, Fan released in [72] EgoVQA, an Egocentric VideoQA dataset which provided the basis to study the importance of such task. In fact, several fields can benefit from advancements in the Egocentric VideoQA task: for example, the industrial training of workers, who may require help in understanding how to perform a certain task given what they see from their own perspective; and the preventive medicine field, where Egocentric VideoQA makes it possible to identify sedentary and nutrition-related behaviours, and help elderly people prevent cognitive and functional decline by letting them review lifelogs [55]. Differently from third-person VideoQA, in the egocentric setting some types of questions can not be posed, such as those pertaining the camera wearer (*e.g.* "what am I wearing?"). Moreover, if the question asks to identify an item the camera wearer is playing with, hands occlusion may partially hide the item, making it hard to recognize.

Data augmentation techniques have proven particularly helpful in several Computer Vision tasks, such as image classification [146]. Not only they can be helpful to avoid

overfitting and thus make the model more general, they can also be used to solve class imbalance in classification problems by synthesizing new samples in the smaller classes [255]. With respect to third-person VideoQA datasets, EgoVQA is a small dataset comprising around 600 question-answer pairs and the same number of clips. Since data augmentation is helpful in such contexts but to the best of our knowledge its effectiveness has never been systematically investigated for the Egocentric VideoQA nor for the VideoQA task, in this Chapter we propose several data augmentation techniques which exploit characteristics given by the task itself. In particular, by exploiting the EgoVQA dataset [72] we show their impact on the final performance obtained by the ST-VQA model [120], which is proven to be effective in the study made by Fan.

The main contributions of this Chapter can be summarized as follows:

- we propose several data augmentation techniques which are purposefully designed for the VideoQA task;
- we show the usefulness of our proposed augmentation techniques on the recently released EgoVQA dataset and try to explain why we observe such improvements;
- we achieve a new state-of-the-art accuracy on the EgoVQA dataset;
- we will release code and pretrained models to support research in this important field.

The rest of the Chapter is organized as follows: in Section 2, we introduce the related work to the topics involved in this Chapter, namely Egocentric VideoQA and data augmentation techniques; in Section 3, we detail both our proposed augmentation techniques and the architecture we use; Section 4 covers the experiments performed and the discussion of the results that we obtained; finally, Section 5 draws the conclusions of this Chapter.

2.2 Related work

In this section we will discuss the work related to the two main topics involved in this Chapter, *i.e.* Video Question Answering, and data augmentation techniques.

2.2.1 VideoQA

Recently, VideoQA has received a lot of attention [72, 73, 79, 120, 308] from researchers both in Computer Vision and NLP fields. Several reasons can be related to this interest, such as the challenges offered by this task and the availability of several datasets, *e.g.* TGIF-QA [120], MSRVTT-QA [308], MSVD-QA [308], ActivityNet-QA [321], and TVQA+ [154], populated by many thousands of examples to learn from.

Modern approaches to this task involve a wide selection of different techniques. Jang et al proposed in [120] to use both temporal attention and spatial attention, in order to learn which frames and which regions in each frame are more important to solve the task. Later on, attention mechanisms have been also used as a cross-modality fusion mechanism [111], and to learn QA-aware representations of both the visual and the textual data [154, 135]. Because of the heterogeneous nature of the appearance and

motion feature which are usually extracted from the video clips, Fan et al [73] also propose to use memory modules, coupled with attention mechanisms, to compute a joint representation of these two types of features. Moreover, to compute the final answer for the given video and question there are multiple approaches. Simpler ones propose to use fully connected networks coupled with non-linear functions [120], but also more complex solutions have been proposed, *e.g.* based on reasoning techniques which exploit multiple steps LSTM-based neural networks [73] or graphs to better encode the relationships between the visual and textual data [111, 124].

Finally, given the multitude of VideoQA datasets, there can also be multiple types of information to exploit. In fact, not only clips, questions, and answers are exploited to solve this task: as an example, TVQA+ [154] also provides subtitles and bounding boxes, by using which it is possible to improve the grounding capabilities of the VideoQA model in both the temporal and the spatial domain.

2.2.2 Egocentric VideoQA

On the other hand, Egocentric VideoQA was a completely unexplored field until very recently, when Fan released the EgovQA dataset in [72]. Yet, considering the recent advancements in several fields of the egocentric vision, such as action recognition and action anticipation [45, 44], Egocentric VideoQA also plays a primary role in the understanding of the complex interactions of the first-person videos.

Both VideoQA and Egocentric VideoQA usually deal with two main types of tasks: the "open-ended" and the "multiple choice" task [72, 120, 308, 321]. Given a visual contents-related question, the difference between the two is due to how the answer is chosen: in the former, an answer set is generated from the most frequent words (*e.g.* top-1000 [308, 321]) in the training set and the model needs to choose the correct answer from it, *i.e.* it is usually treated as a multi-class classification problem; in the latter the model needs to select the correct answer from a small pool of candidate answers (*e.g.* five choices [72, 120]), which are usually different for every question. In this Chapter we focus on the multiple choice task.

Together with the release of the EgoVQA dataset, Fan also provided in [72] a baseline made of four models borrowed from the VideoQA literature [73, 79, 120]. These models use the same backbone, which consists in a frozen, pretrained VGG-16 [259] to extract the frame-level features; a frozen, pretrained C3D [276] to extract the video-level features; and a pretrained GloVe [214] to compute the word embeddings. The four models can be seen as extensions of a basic encoder-decoder architecture (referred to as "ST-VQA without attention" in [72]): "ST-VQA with attention" is based on [120] and uses a temporal attention module to attend to the most important frames in the input clip; "CoMem" is based on [79] and involves the usage of two memory layers to generate attention cues starting from both the motion and appearance features; and finally "HME-VQA" [73] uses two heterogeneous memory layers and a multi-step reasoning module. In [72], Fan shows that these four models achieve similar performance despite the introduction of several cutting-edge modules. Because of this reason and because of its simplicity, in this Chapter we focus on the "ST-VQA with attention" model.

2.2.3 Data augmentation techniques

Several Computer Vision tasks, such as image classification [146] and handwritten digit classification [149], have seen great improvements by exploiting data augmentation techniques, through which the size of the training set can be expanded artificially by several orders of magnitude. This leads to models which are far less susceptible to overfitting and more prone to give better results during the testing phase.

Whereas papers about first- or third-person VideoQA never mention any augmentation technique, in the VisualQA task, which deals with question-answering over images, there are some papers which try to tackle this opportunity by exploiting template-based models or generative approaches. Using a semantic tuple extraction pipeline, Mahendru et al [187] extract from each question a premise, *i.e.* a tuple made of either an object, or an object and an attribute, or two objects and a relation between them, from which new question-answer pairs are constructed using previously built templates. Kafle et al [130] proposes two techniques. The first is a template-based method which exploits the COCO dataset [170] and its segmentation annotations to generate new question-answer pairs of four different types, exploiting several different templates for each type. The second approach is based on sequentially generating new questions (and related answers) by conditioning an LSTM-based network on the image features from the "VQA dataset" [5]. Both these methods focus on creating new question-answer pairs for the same image, either by exploiting purely linguistic aspects or by using visual information to better guide the generation. Yet, as the authors report in the respective papers, these methods although reliable are not error-free [187, 130]; on the other hand, our proposed techniques are simple yet effective and they do not raise issues. In particular, in this Chapter we focus on exploiting both the visual and the textual data, although we do not create new questions: two of our techniques create new candidate answers for the same question to strengthen the understanding of the concepts contained in the question and to better distinguish between the correct and the wrong answers, and the other technique creates "new" clips by horizontally flipping the frames and consistently updating both the question and the candidate answers accordingly.

2.3 Methodology

In this section we will introduce and describe the proposed augmentation techniques. Moreover, we will also discuss and describe the model used in this Chapter, which is called "ST-VQA" and was initially introduced by Jang et al in [120].

2.3.1 Augmentation techniques

Following the work made in [72], we are working on the multiple choice setting. For each video and question five candidate answers are provided, of which only one is correct. The wrong answers are randomly sampled from a candidate pool based on the question type, *i.e.* if the question requires to recognize an action, the five candidate answers (both the right one and the four wrong) are actions. By doing so, the model is encouraged to understand the visual contents in order to reply to the question, avoiding the exploitation of pure textual information (*e.g.* exploiting the question type to filter out some of the candidate answers).

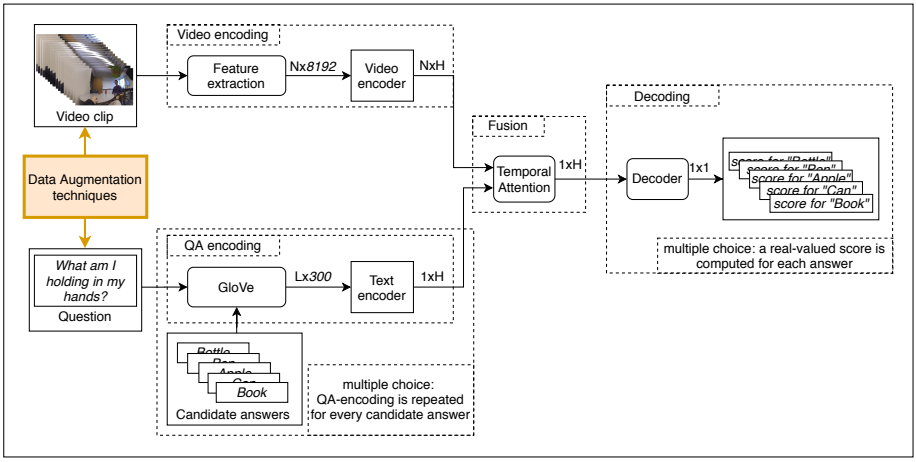


Figure 2.1: Model used in this Chapter. The feature extraction module is made of VGG and C3D, whose output consists of N frames and 8192 features. The output of GloVe consists in fixed-length vectors (*i.e.* embeddings) of embedding size $E = 300$. The Video Encoder and the Text Encoder have a similar structure, but whereas the output of the former is a sequence (length N) of hidden states, the output of the latter is a single hidden state. The Decoder outputs a single real-valued score for each candidate answer.

We propose to use three simple augmentation techniques designed for the VideoQA task and which exploit the multiple choice setting: resampling, mirroring, and horizontal flip. This is not only helpful when dealing with the overfitting, but can also give the model a better understanding of what the questions is asking for and make the model more robust to variations in the input frames.

Resampling

Given a question Q, in the multiple choice setting a handful (*e.g.* 5 choices [72], [120]) of candidate answers are considered. The first technique consists in fixing the correct answer and then randomly *resampling* the wrong ones. By doing so, using the same video and question, we can show the model several more examples of what is *not* the correct answer. This should give the model the ability to better distinguish what the question is and is not asking for. An example is shown in Figure 2.2.

Considering that the amount of possible tuples of wrong answers is exponentially big, we are restricting the pool of wrong answers to those pertaining the same question type of Q. Moreover, we are not considering all the possible tuples in the pool.

Mirroring

Given a question, it may be that the correct answer in the rows of the dataset is often placed in the same position. This can create biases in the model which may tend to prefer an answer simply based on its position (w.r.t. the order of the candidates). To

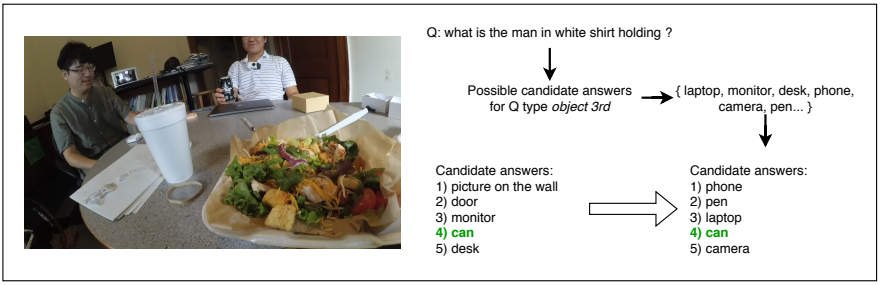


Figure 2.2: Example of the "resampling" technique applied to a video clip in the EgoVQA dataset.

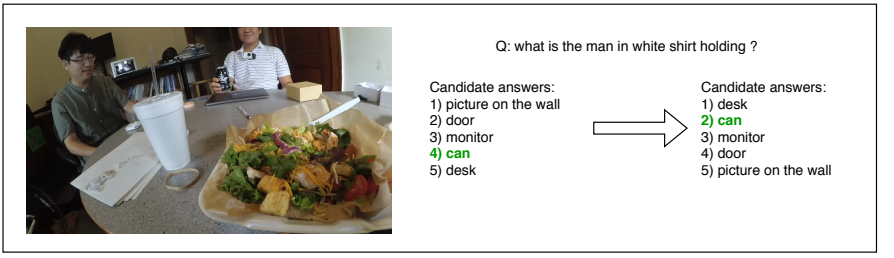


Figure 2.3: Example of the "mirroring" technique applied to a video clip in the EgoVQA dataset.

relieve some of this bias we propose the *mirroring* technique, which consists in simply adding a row to the dataset where the order of the candidate answers (and the label value) is mirrored. An example is shown in Figure 2.3.

Horizontal flip

One of the most common image data augmentation techniques consists in horizontally flipping the images, which often improves the model performance thanks to the availability of newly created images which are taken both from the left and from the right. This technique may prove useful in a VideoQA setting as well, but it should not be applied lightly because it is a non-label preserving transformation: horizontally flipping the considered frame means that an object which was on the left side of the frame appears on the right side after the transformation, and viceversa, eventually creating wrong labels if not updated correctly. Thus, when flipping the frames in the video clip both the question and the candidate answers likely need to be updated (*e.g.* Figure 2.4).

2.3.2 QA encoding: Word embedding and Text Encoder

As shown in Fig. 2.1, it can be seen as made of four blocks: Question-Answer (QA) Encoding, Video Encoding, Fusion, and Decoding. To compute the word embeddings for the question and the answers, we consider GloVe [214], pretrained on the Common

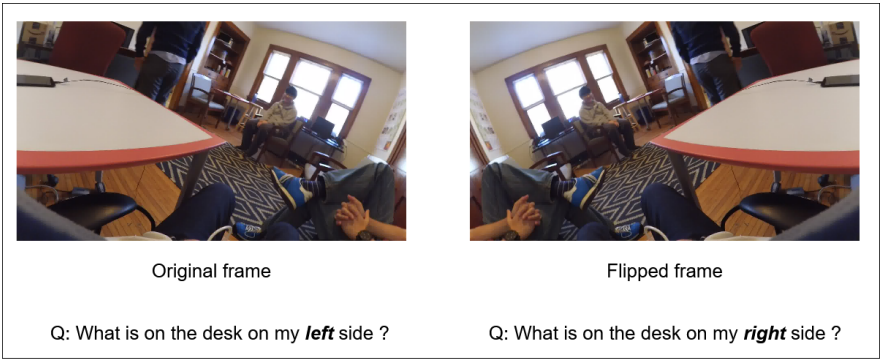


Figure 2.4: Example of the "horizontal flip" technique applied to a frame of a video clip in the EgoVQA dataset.

Crawl dataset[1], which outputs a vector of size $E = 300$ for each word in both the question and the answers. Since GloVe is not contextual, question and answer can be given in input to the model either separately or jointly obtaining the same embedding.

First of all, the question and the candidate answer are tokenized, *i.e.* they are split in sub-word tokens and then each of them receives an identifier, based on the vocabulary used by GloVe. Let $q_1 \dots q_m$ and $a_1 \dots a_n$ be the sequence of m words of the question and n words of (one of the candidate) answer, and let $L = m + n$. Thus, let $\phi_q \in \mathbb{R}^{m \times E}$ be the question embedding, and $\phi_a \in \mathbb{R}^{n \times E}$ be the answer embedding. The final question-answer embedding is computed as their concatenation, *i.e.* $\phi_w = [\phi_q, \phi_a] \in \mathbb{R}^{L \times E}$.

Then the Text Encoder, consisting of two stacked LSTM networks, is applied to ϕ_w. By concatenating the *last* hidden state of both the LSTM networks we obtain the encoded textual features $\epsilon_w \in \mathbb{R}^{1 \times H}$, where H is the hidden size.

2.3.3 Video Encoding

From each input video clip, both motion and appearance features are obtained in the Video Encoding module. In particular, the appearance features are computed as the $fc7$ activations ($\phi_a \in \mathbb{R}^{N \times 4,096}$) extracted from a frozen VGG-16 [259], pretrained on ImageNet [146]. We use VGG because we want to keep the spatial information extracted by the convolutional layers, which would be otherwise lost in deeper networks, such as ResNet [98], which exploit a global pooling layer before the FC layers. Similarly, the motion features are computed as the $fc7$ activations ($\phi_m \in \mathbb{R}^{N \times 4,096}$), extracted from a frozen C3D [276], pretrained on Sports1M [132] and fine-tuned on UCF101 [265]. Finally we concatenate these features and obtain a feature vector $\phi_{a,m} \in \mathbb{R}^{N \times 8,192}$, which is then encoded by a Video Encoder module, consisting of two stacked LSTM networks. The only difference between the Text and Video Encoder module is that the output ϵ_v of the latter consists in the concatenation of the full sequence of hidden states from both the networks, and not only the last hidden state. Thus $\epsilon_v \in \mathbb{R}^{N \times H}$ represents the encoded video features.

[1]The Common Crawl dataset is available at http://commoncrawl.org

2.3.4 Fusion

Depending on the question (and eventually the candidate answer), a frame may be more or less relevant. To try and exploit this information, the fusion block consists of a temporal attention module that lets the model learn automatically which frames are more important based on both the encoded video features and the textual features. In particular, the temporal attention module is based on the works by Bahdanau et al [11] and by Hori et al [106]. It receives in input the encoded video features $\epsilon_v \in \mathbb{R}^{N \times H}$ and the encoded textual features $\epsilon_w \in \mathbb{R}^{1 \times H}$, and can be described by the following equations:

$$\omega_s = tanh(\epsilon_v W_v + \epsilon_w W_w + b_s) W_s \tag{2.1}$$

$$\alpha = softmax(\omega_s) \tag{2.2}$$

$$\omega_a = \mathbb{1}(\alpha \circ \epsilon_v) \tag{2.3}$$

where $W_v, W_w \in \mathbb{R}^{H \times h}$, $W_s \in \mathbb{R}^{h \times 1}$ are learnable weight matrices, $b_s \in \mathbb{R}^{1 \times h}$ is a learnable bias. $\circ$ represents the element-wise multiplication operator. By means of $\alpha \in \mathbb{R}^{N \times 1}$ we aim at capturing the importance of each of the visual feature vectors and their interaction with the textual information. The output of the Fusion module is a feature vector $\omega_a \in \mathbb{R}^{1 \times H}$, which is obtained by accumulating the N attended visual feature vectors through Eq. 2.3, which uses a row of ones ($\mathbb{1}^{1 \times N}$) and is equivalent to $\omega_a = \sum_{i=1}^{N} (\alpha_i \circ \epsilon_{v,i})$.

2.3.5 Decoding

Finally, the decoding step considers both the attended features computed by the Fusion block and the encoded textual features, as proposed by Fan [72]. In our multiple choice setting, the decoding is performed five times, *i.e.* for each Q-A pair, with different textual features producing five different scores, one per candidate answer. It can be described by the following equations:

$$d_f = tanh(\omega_a W_a + b_a) \tag{2.4}$$

$$d_r = (d_f \circ \epsilon_w) W_d + b_d \tag{2.5}$$

where $W_a \in \mathbb{R}^{H \times H}$ and $W_d \in \mathbb{R}^{H \times 1}$ are learnable weight matrices, $b_a \in \mathbb{R}^{1 \times H}$ and $b_d \in \mathbb{R}$ are learnable biases, $d_f \in \mathbb{R}^{1 \times H}$, $d_r \in \mathbb{R}$. d_r can be seen as the score obtained by testing a specific candidate answer (out of the five possible choices related to the given question).

2.3.6 Loss function

The model is trained using a pairwise hinge loss, as is done in [72, 120]. The loss function can be defined as follows:

$$\mathcal{L}_{c,r} = \begin{cases} 0 & \text{if } c = r \\ max(0, 1 + s_n - s_p) & \text{if } c \neq r \end{cases} \tag{2.6}$$

Figure 2.5: Samples of video clips, questions, and candidate answers from the EgoVQA dataset.

where s_n and s_p are respectively the scores d_f computed by the decoder for the choice $c \in \{1, \ldots, 5\}$ and the right answer r. As described in Sec. 2.3.5, our model is producing a score (real number) for each input tuple (V, Q, A), and the score for the correct answer should be the highest among the five candidates; by using the hinge loss we enforce a margin between the score s_n obtained by the wrong answer and the score s_p obtained by the correct one, aiming at obtaining $s_p \geq s_n + 1$.

Finally we compute the loss as follows:

$$\mathcal{L} = \sum_{q \in \mathcal{Q}} \sum_{c=1}^{5} \mathcal{L}_{c,r} \tag{2.7}$$

Here $\mathcal{Q}$ is the set of the questions, and r is the right answer for the question q.

2.4 Results

In this section we briefly describe the dataset used to perform the experiments, and we discuss both the overall results and the per question type results.

2.4.1 EgoVQA dataset

The EgoVQA dataset was recently presented by Fan [72]. It features more than 600 QA pairs and the same number of clips, which are 20-100 seconds long and are obtained by 16 egocentric videos (5-10 minutes long) based on 8 different scenarios. An example of these egocentric videos and QA pairs can be seen in Fig. 2.5. The questions can be grouped in eight major types and they are described in Table 2.1.

2.4.2 Implementation details

In our setting, we fixed $H = 512$ and $h = 256$. To optimize the parameters we used the Adam [141] optimizer with a fixed learning rate of 10^{-3} and a batch size of 8. To

Table 2.1: Description of the question types available for testing in the EgoVQA dataset.

Code	Question type	Quantity	Example
Act_{1st}	Action 1st	67	"what am I doing"
Act_{3rd}	Action 3rd	108	"what is the man in red clothes doing"
Obj_{1st}	Object 1st	54	"what am I holding in my hands"
Obj_{3rd}	Object 3rd	86	"what is placed on the desk"
Who_{1st}	Who 1st	13	"who am I talking with"
Who_{3rd}	Who 3rd	63	"who is eating salad"
Cnt	Count	64	"how many people am I talking with"
Col	Color	31	"what is the color of the toy in my hands"

Table 2.2: Performance computed for each of the splits, obtained by applying the proposed techniques on the EgoVQA dataset.

Augmentation	Accuracy (%) on split			
	0	1	2	Avg
ST-VQA [120]	31.82	37.57	27.27	32.22
+ mirroring	32.58	40.46	23.53	32.19
+ resampling	26.52	28.90	29.41	28.28
+ mirroring	37.88	36.42	**30.48**	34.93
+ horizontal-flip	34.09	41.62	25.13	33.61
+ resampling	37.12	35.26	25.67	32.68
+ mirroring	**40.91**	**43.35**	28.88	**37.71**

implement our solution we used Python 2.7, Numpy 1.16, and PyTorch 1.4. A PyTorch implementation will be made available to further boost the research in this important area at https://github.com/aranciokov/EgoVQA-DataAug.

2.4.3 Discussion of the results

Table 2.2 shows the results obtained for each of the three splits proposed in [72] by applying, with different combinations, our proposed augmentation techniques. Table 2.3 shows the results based on the question type, whose details (and codes, such as "Act_{1st}" and "Act_{3rd}") are defined in Table 2.1.

Overall it can be seen that, when used in conjunction, the proposed augmentation techniques help improving the performance obtained by the considered model.

Looking at the results per question type, it is possible to notice that:

- the "resampling" technique is particularly helpful when it comes to counting objects ("Cnt") and identifying objects used by actors in front of the camera wearer ("Obj_{3rd}");
- the "mirroring" technique shows sensible improvements during the identification of actors, both when the camera wearer is interacting with them ("Who_{1st}") and when they are performing certain actions in front of the camera wearer itself ("Who_{3rd}").

Table 2.3: Per-question type results obtained by applying the proposed techniques on the EgoVQA dataset.

Augmentation	Question type accuracy (%)							
	Act_{1st}	Act_{3rd}	Obj_{1st}	Obj_{3rd}	Who_{1st}	Who_{3rd}	Cnt	Col
ST-VQA [120]	28.36	30.56	31.48	31.40	46.15	34.92	35.94	32.26
+ mirroring	26.87	33.33	35.19	27.91	**53.85**	46.03	26.56	19.35
+ resampling	26.87	26.85	20.37	37.21	15.38	23.81	**43.75**	16.13
+ mirroring	25.37	34.26	25.93	41.86	15.38	**47.62**	42.19	19.35
+ horizontal-flip	**40.30**	36.11	**42.59**	25.58	38.46	28.57	26.56	**41.94**
+ resampling	34.33	37.96	22.22	33.72	15.38	33.33	29.69	29.03
+ mirroring	31.34	**39.81**	22.22	**44.19**	15.38	**47.62**	37.50	38.71

- the "horizontal flip" technique is especially helpful when the model needs to identify the actions performed by ("Act_{1st}") and the objects over which the action is performed by the camera wearer ("Obj_{1st}"). Moreover, it gives the model a great boost in recognizing colors ("Col").

Both in the questions of type "Cnt" and "Obj_{3rd}" the model is required to recognize an object: in fact, whereas in the latter the model needs to identify an object by distinguishing among the five candidates, the former also requires the model to understand what such object is in order to count how many times it occurs in the scene. It is interesting to notice that in our "resampling" technique we are not augmenting the questions of type "Cnt", because the only five possible candidate answers in the dataset for such question type are the numbers from "one" to "five". Thus, since we are able to observe this improvement in both these question types, it likely implies that such data augmentation technique helps the model to better distinguish among the different objects available in the dataset because it provides several more examples where the model needs to understand which object is the right one among several (wrong) candidate answers.

In the case of the question types "Who_{3rd}" and "Act_{3rd}" the improved performance may be due to two aspects: first of all, since they both require to recognize actions performed by actors in front of the camera wearer, the accuracy gain obtained in one type transfers (to some extent) to the other type, and viceversa; and then to the "mirroring" technique, since it is possible to observe that in the training set there is a bias in both question types towards one of the last two labels. In particular, over the three training splits, the last candidate answer is the correct one 60 times over 203 questions (29.55%) of type "Act_{3rd}", whereas "Who_{3rd}" counts 27 instances of the second-to-last candidate answer over a total of 77 questions (35.06%). Using the "mirroring" technique it is thus possible to reduce this bias, making the model more robust. It is interesting to notice that in both these question types, the addition of the "horizontal flip" technique gives a further boost in the accuracy of the model. This is likely related to the fact that several questions in the training data also contain a positional information ("left", "right") of the actor involved: in particular, for the type "Act_{3rd}" there are respectively 16 and 20 questions mentioning "left" or "right" over a total of 203 questions, whereas for "Who_{3rd}" there are respectively 2 and 3 over a total of 77.

In the question type "Who_{1st}" we can observe a sensible improvement with the

"mirroring" technique. Although the reason are likely similar (considering that the first candidate answer is the right one 11 times over 20 instances for "Who_{1st}"), we prefer not to make any conclusive claim given that there is only a total of 20 instances in the training set and 13 in the testing set.

The "horizontal flip" technique shines when asked to recognize which object the camera wearer is interacting with ("Obj_{1st}") and to identify which action ("Act_{1st}") is performed by the camera wearer itself. The improvement over the former question type may be explained by the fact that several of its questions in the training data involve a positional information: in particular, there are respectively 18 and 9 questions containing "left" or "right" over a total of 86 questions. On the other hand, the great improvement in the latter question type (whose questions are almost all of the form "what am I doing") is likely justifiable by the greater amount of different visual data available for training.

Finally, among our proposed techniques, only the "horizontal flip" seems to cope well with "*Col*" questions. This question type is particularly tough because it requires the model to recognize the object which the question is referring to, the action which is performed over the object (35/49 total instances), and sometimes even the colors of the clothes of the actor (*e.g.* "what is the color of the cup held by the man in *black* jacket", 13/49 total instances). First of all, the "mirroring" technique does not help: the training split are slightly biased towards the first and the last labels (respectively, 10 and 16 over 49 instances), meaning that in this case the proposed technique does not resolve the bias towards these two labels. Secondly, our "resampling" technique is not helping because there are only six unique colors in the dataset, thus it is not creating enough new rows. Thirdly, only 2 over a total of 54 questions of this type in the training data contain "left" or "right", likely implying that the improvement obtained by the "horizontal flip" technique is due to having more visual data which forces the model to better understand where to look for the object targeted by the question.

2.5 Conclusion

Egocentric VideoQA is a task introduced recently in [72] which specializes the VideoQA task in an egocentric setting. It is a challenging task where a model needs to understand both the visual and the textual content of the question, and then needs to jointly attend to both of them in order to produce a coherent answer. In this Chapter we proposed several data augmentation techniques purposefully designed for the VideoQA task. The "mirroring" technique tries to partially remove the ordering bias in the multiple choice setting. The "resampling" technique exploits the training dataset to create new question-answer pairs by substituting the wrong candidate answers with different candidates from the same question type, in order to feed the network with more examples of what is not the target of the question. Finally, the "horizontal-flip" technique exploits both the visual and the textual content of each row in the dataset, and aims at giving the model the ability to differentiate between "left" and "right". To show the effectiveness of these techniques, we tested them on the recently released EgoVQA dataset and showed that we are able to achieve a sensible improvement (+5.5%) in the accuracy of the model.

As a future work, we are both considering to explore our proposed augmentation techniques with other architectures, such as the HME-VQA model [73], and to replicate these experiments in third-person VideoQA datasets. Moreover, we are considering several different augmentation techniques that deal with the linguistic aspects and the visual information both separately and jointly. In particular, we think that considering them jointly is of most interest because of the inherent characteristics of the problem setting, which requires the model to understand both linguistic and visual clues together. A purely linguistics technique which we plan to explore consists in a variation of the "mirroring" technique which *permutes* the candidate answers instead of simply mirroring them: this should reduce the ordering bias in all the possible situations, even those where the "mirroring" technique is weaker. The "resampling" technique could be further improved by picking candidates which are grounded in the visuals or which pose a greater challenge to the model, instead of choosing them randomly: for instance, if the question is about "what am I eating", then the video may show both the meal of the camera wearer and that of other people, therefore by picking the latter as wrong candidates requires the model to avoid linguistic biases and be more careful at the visual content. Another technique, although focusing solely on the visual data consists in applying small rotations to the video clips, considering that the egocentric camera may not be aligned at all times due to the camera wearer moving in the scenario. Finally, reversing the video clips and updating both question and the candidate answers accordingly (*e.g.* by "reversing" the name of the actions performed in the video clip) may give the model a more clear understanding of the actions, while better exploiting the sequential nature of the visual data.

3

A Feature-space Multimodal Data Augmentation Technique for Text-video Retrieval

3.1 Introduction

The amount of user-generated video content uploaded to the Internet every minute is ever increasing, leading to more than 500 hours of content uploaded to YouTube every minute, as of February 2020 [25]. Finding the relevant videos for a given query requires a mix of computer vision and natural language processing techniques, placing this problem at the intersection of the two communities. In particular, the text-to-video retrieval task encompasses this objective by requiring to sort all the videos based on their semantic closeness to the input query. Another task, which is similar to text-to-video retrieval

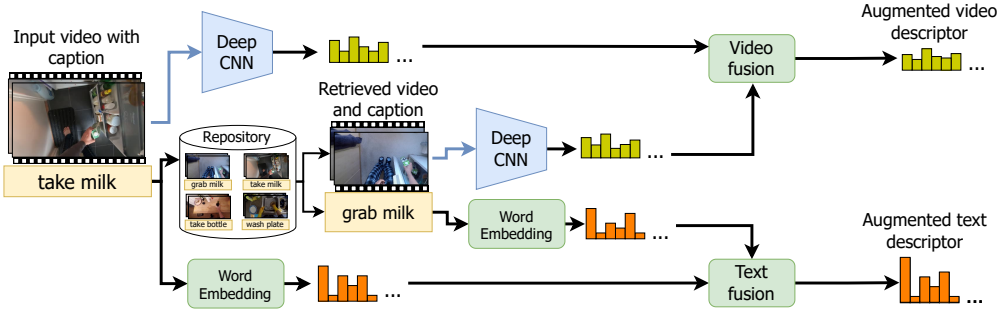


Figure 3.1: Overview of the proposed multimodal data augmentation technique working on latent representations.

and is used to holistically evaluate a method, is the video-to-text retrieval task, which switches the role of video and query. In general, with the term text-video retrieval both tasks are considered and, given its cross-modal nature, it involves both visual and textual understanding.

Recently, deep learning techniques were used to automatically extract features from the multimodal data and learn how to solve this task, showing their potential and achieving impressive results [288, 40, 212]. However, a significant limitation in the success of these techniques is represented by the huge amount of annotated data required to perform the training of a deep learning model. To this end, large amounts of data were collected through crowdsourcing platforms where human efforts are required to carefully annotate the data, leading to tedious tasks for the annotators and huge costs for the dataset collectors. Examples of large scale datasets obtained with this approach include MSR-VTT [310] and VATEX [289]. To reduce the costs of the collection, the scientific community mainly investigated two automatic solutions: web scraping and data augmentation. In the former, the extraction of visual content from the Internet and the related annotation is performed automatically, for instance with speech recognition [195], alternative texts [12], or by leveraging hashtags [84]. While this approach leads to possibly huge and rich datasets, the annotations are often noisy and it is difficult to guarantee the quality of the annotations. On the other hand, data augmentation techniques are often used to artificially increase the size of a dataset by leveraging the already available annotated samples: new samples can be obtained by applying label-preserving techniques, hence providing semantically coherent data and avoiding the noise. Indeed, these techniques have shown a great potential in many fields, both from the vision community, such as classification [146, 312, 14, 285] and detection [341, 228], and the language processing community, such as text summarization [209, 66] and text classification [294, 147]. Although augmentation was applied to visual question answering [252, 292] and image captioning [41, 281], these techniques are less explored for text-video retrieval. To address this shortcoming, we investigate the application of augmentation techniques and propose an augmentation technique for text-video retrieval which exploits multimodal information (visual and textual). In particular, our video augmentation strategy creates a new augmented video by mixing the visual features of two samples from the same class ('Video fusion' in Fig.3.1), therefore leveraging the high level concepts automatically extracted from the deeper layers of a CNN-based backbone. This is achieved by performing our augmentation in the feature space, as opposed to common transformations, such as the geometric and color space transformations used for images, which are applied on the raw data [146]. In fact, working in the feature space raises three additional advantages: the same technique can be applied to data coming from different modalities, for instance on both video and text as we show in this Chapter, without requiring considerable changes which, on the other hand, are likely required when trying to apply a technique defined on one modality (*e.g.*, replacing a word with a synonym) on a completely different modality (*e.g.*, on video); it does not rely on the availability of the original videos or frames, which are more difficult to share and are not always shareable due to privacy or copyright issues, *e.g.* more than 20% of the original videos of MSR-VTT were reported to be removed from YouTube [194], whereas all the videos of MovieQA [273] faced copyright issues; and finally it can be applied on pre-extracted features, making it overall less time- and resource-demanding. The augmented

caption for the abovementioned video is also created by following the same principle ('Text fusion' in Fig.3.1), showing the general applicability of our technique to multiple types of media. Finally, to validate our approach, multiple experiments are performed on the recently released EPIC-Kitchens-100 dataset [45]. These experiments include: multiple ablation studies to demonstrate the effectiveness of our strategy and to motivate the design choices; several comparisons to augmentation techniques inspired from the literature; and finally, to give additional evidence of the usefulness of our method, we observe further improvements when our proposed technique is integrated with a state-of-the-art model. To support reproducibility, code and pretrained models are made publicly available on Github at https://github.com/aranciokov/FSMMDA_VideoRetrieval.

We organize the Chapter as follows. In Section 3.2 we perform a literature review and contextualize our work into it. Then, in Section 3.3 we described in detail the proposed technique. Several ablation studies and experiments are performed and discussed in Section 3.4, whereas in Section 3.5 we conclude this Chapter.

3.2 Related work

Since this Chapter focuses on the exploration of data augmentation techniques for the text-video retrieval task, we reserve Section 3.2.1 for the augmentation techniques which were proposed in vision and language fields. Then, in Section 3.2.2 we briefly describe recent modeling approaches in the text-video retrieval community.

3.2.1 Data augmentation techniques

Data augmentation techniques are widely used in computer vision because they allow creating new data points. Several techniques working on the raw data were proposed. Standard geometric or color space transformations, such as rescaling, rotation, variations in the brightness, *etc* were used in multiple contexts related to images [146, 14] and, by applying the same transformations in a frame-by-frame fashion, also to videos [117, 239]. Specific techniques were introduced to leverage the temporal nature of videos, including temporal subsampling [285], inversion of the sequence of frames [156], or the replacement of part of the video with a different cuboid [325]. Furthermore, as described in a recent survey by Cauli et al., generative models [1, 293] and simulation programs [110, 113] were also used to generate new data [24].

At the same time, in the natural language processing community several interesting techniques were proposed, which can be categorized into symbolic and neural techniques as explained in the comprehensive survey [256] by Shorten et al. A key difference between the two categories is represented by the usage of additional neural models in the latter. Symbolic augmentations work on the raw words or sentences and include random word insertion, deletion, and swapping [294], synonym replacement [294, 287], passivization, and subject-object inversion [191, 199]. Neural augmentation rely on neural models to augment the available textual data, for example by leveraging back-translation [217, 178] or generative models [302].

Some of these techniques were also extended or adapted for tasks at the intersection of the vision and language communities. Rephrasings of questions and a cycle-consistency loss were introduced by Shah et al. to make a more robust model for visual question

answering [252], whereas Wang et al. used a generative model to generate questions and answers [292]. To alleviate overfitting in image captioning, Wang et al. [281] performed cropping, rescaling, and mirroring on images, whereas Cui et al. [41] created image-text pairs used as negative examples by replacing or permuting words or full sentences. A few recent works were also proposed for text-image retrieval. Wang et al. generated new captions from the images with a pre-trained image captioning model [286]. Zhan et al. used a 'cut-and-paste' technique to vary the background features of product images [328].

While all these techniques prove to be powerful and help learning richer representations, they are based on the raw data and require their availability, which may be difficult to share and even not shareable due to privacy or copyright issues, *e.g.* clips from movies or TV series. Conversely, data augmentation techniques working at the feature level are less computationally intensive and can provide considerable improvements. Examples of these techniques either work on one vector at a time, *e.g.* by using noising techniques [307, 34], or multiple, for instance by interpolating two samples from the same class [174, 147] or by varying one in terms of the center of its class [34]. Augmentation techniques working in the latent space were used to augment images [174, 34] and text [307, 147]. Nonetheless, these techniques are less explored in the video community. In particular, Dong et al. performed data augmentation in the feature space by temporally downsampling the sequences and perturbing the video features with noise injection [56, 58].

To the best of our knowledge, data augmentation techniques, both on the raw data and in the feature space, were not used in the text-video retrieval field.

3.2.2 Text-video retrieval

Text-video retrieval is a cross-modal task comprising two symmetric sub-tasks, text-to-video and video-to-text retrieval, depending on which modality is used to form the query and the ranking list. An approach which is commonly used consists in learning a textual-visual embedding space by means of a contrastive loss [193, 92, 247, 93]. Generally, this means that the embeddings of each video and caption pair (the 'positive' examples) in the dataset are extracted and their similarity is maximized; the similarity of pairs of video and caption which are not associated in the dataset (called 'negative' examples) may be also considered for the loss.

Many different methods were proposed for the text-video retrieval task. Several authors leveraged the availability of very large scale datasets to perform vision and language pretraining [195, 151, 173], but these methods often are not designed for the task at hand and are computationally expensive. Differently from them, learning how to aggregate the multiple representations available was explored for both the visual [288, 175, 76] and textual data [40, 164]. Finally, instead of working with global features, several authors shifted the attention to the alignment of local components. Wray et al. learned multiple embedding spaces based on part-of-speech [299]. Chen et al. extracted semantic role graphs of the captions and aligned each node to learned representations of the clips [27]. On a similar note, Jin et al. computed a graph representation of the video in three levels and aligned them to local components of the sentences [127].

3.3 Feature-space multimodal data augmentation

Learning a model for the text-video retrieval task often involves two neural networks to compute the two representations of the input video and related caption. Then, the similarity of these representations is increased, requiring the preceding networks to adjust their weights in order to compute a similar representation for both the video and the caption. By doing so, the input caption may be at the top of the ranked list given its video, and vice versa. Yet, multiple captions (and videos) may be equally relevant and thus rightfully placed at the same rank. Therefore, we propose a multimodal data augmentation technique which creates new representations for videos and captions by mixing those which share similar semantics. In particular, our augmentation is performed in the feature space, leading to multiple advantages: by working on the features extracted from the deeper layers of the backbones, the augmented representations encompass high level concepts, as opposed to the low level characteristics used by techniques working on raw data; the technique is easy to extend to different modalities, since it works on latent representations; by only requiring pre-extracted features to be shared, there are less concerns regarding the shareability and availability of the original raw data; less computational resources are needed to perform the augmentation, as the feature extraction from the raw data can be performed offline.

As an example which further motivates the proposed technique, let v_1 and v_2 be two videos showing different people while rinsing a fork with running water. To describe this action, verbs such as "cleaning", "washing", or "rinsing" may be used, whereas the fork may also be pointed with more general ("cutlery" or "silverware") or more specific terms ("fork with 3 tines" or "stainless steel fork"). All these captions share similar semantics with only small variations, which may be captured by the high level features automatically extracted from a deep neural network. Therefore, these features may be reused and mixed to obtain a new representation for a caption which shares similar semantics as the original ones. Similarly, we may treat v_1 and v_2 as interchangeable and, even more interestingly, possibly mixable.

In the following Sections 3.3.1 and 3.3.2 we describe in detail how to generate new clip and new caption features from the available information. An overview of the whole process is shown in Algorithm 1.

3.3.1 Generating a new clip from same-class samples interpolation

First of all, we define two selection criteria, ϕ_V and ϕ_N, which identify compatible videos with respect to the action performed or the object with which the interaction happens. This means that if a is an action and o is an object, then $\phi_V(a)$ and $\phi_N(o)$ are sets of videos which are representatives of a and o. Note that this criterion may lead to far too much variance: for instance, $\phi_V(\text{take})$ may contain videos about taking a fork from the cupboard, or picking it up from the table, but a video showing someone taking a slice of pizza would also be identified as compatible. While this may gather many more videos, both highly or minimally relevant, and help pushing them all at the top of the ranked list, it may also raise additional confusion and lower precision. Therefore, we further constrain ϕ_V and ϕ_N by keeping them bound to both the entities and the actions of the

Algorithm 1 Algorithm used to perform the augmentation at *training* time.

1: **Input**: video v, caption q
2: **Output**: eventually augmented descriptors $\overline{v}$ and $\overline{q}$
3: $\overline{v} \leftarrow f(v)$, $\overline{q} \leftarrow g(q)$ ▷ v and q are embedded
4: $p \sim U(0, 100)$
5: **if** $p > (1 - \chi) \cdot 100$ **then** ▷ If we perform the augmentation
6: $N\_or\_V \sim U(0, 1)$ ▷ On actions or entities?
7: **if** $N\_or\_V == 0$ **then** ▷ On entities
8: $\phi \leftarrow \phi_N$, $\psi \leftarrow \psi_N$, $\texttt{fn} \leftarrow \texttt{ent}$ ▷ Set the correct ϕ, ψ, and `fn` functions
9: **else** ▷ On actions
10: $\phi \leftarrow \phi_V$, $\psi \leftarrow \psi_V$, $\texttt{fn} \leftarrow \texttt{act}$
11: **end if**
12: $c \leftarrow c \sim \texttt{fn}(v)$ ▷ Sample an action/entity from v
13: $w \leftarrow w \sim \phi(c, v)$ ▷ Sample a substitute video
14: $\overline{w} \leftarrow f(w)$ ▷ w is embedded
15: $\overline{v} \leftarrow \mu(\overline{v}, \overline{w})$ ▷ Create the new video
16: $t \leftarrow t \sim \texttt{fn}(q)$ ▷ Sample an action/entity from q
17: $d \leftarrow d \sim \psi(t, q)$ ▷ Sample a substitute from the candidates
18: $\overline{d} \leftarrow g(d)$ ▷ d is embedded
19: $\overline{q} \leftarrow \rho(\overline{q}, \overline{d})$ ▷ Create the new caption
20: **end if**
21: **return** $\overline{v}$, $\overline{q}$

video:

$$\phi_V(a, v) = \{w \,|\, a \in \texttt{act}(w) \ \wedge \ (\texttt{ent}(v) \cap \texttt{ent}(w)) \neq \emptyset\} \tag{3.1}$$

$$\phi_N(o, v) = \{w \,|\, o \in \texttt{ent}(w) \ \wedge \ (\texttt{act}(v) \cap \texttt{act}(w)) \neq \emptyset\} \tag{3.2}$$

where w represents a sampled video. Here `act` and `ent` are functions used to extract the semantic classes for the actions and entities in the corresponding captions. As an example, `act`(pick a slice of pizza) will be a set containing the identifier of the class for 'pick', and `ent`(pick a slice of pizza) will contain the one for 'slice of pizza'. To obtain the functions `act` and `ent`, a pipeline made of a part-of-speech tagger and a lexical database (*e.g.* WordNet [198]) can be used. If each video is paired with multiple captions, the semantic classes for it may include those which are shared among multiple captions, as in Wray et al. [298].

As shown in Algorithm 1, we decide whether or not to perform the augmentation of a given sample with chance χ (steps 4-5), therefore using both original and augmented samples during training. Then, the choice between actions and entities is taken with uniform chance (step 6) and the corresponding criteria are selected (steps 7-11). To create the augmented sample, two more variables need to be sampled: the semantic class (action or entity) which will be used to find a compatible w, and the actual sampling of w from all the possible candidates found through ϕ (steps 12-13). Finally, a new "virtual" member of the same class as v and w is obtained by extracting their vectorial representations $\overline{v}$ and $\overline{w}$ with a function f (steps 3 and 14) and combining them with

$\mu(\overline{v}, \overline{w})$ ('Video fusion' in Fig.3.1). In our method, we define μ as a linear interpolation of v and w, by implementing it as:

$$\mu(\overline{v}, \overline{w}) = \lambda \cdot \overline{v} + (1 - \lambda) \cdot \overline{w} \tag{3.3}$$

and by sampling λ from a Beta distribution with both parameters set to 1, *i.e.* $\lambda \sim \beta(1, 1)$, inspired by Mixup [333]. By doing so, $\mu(\overline{v}, \overline{w})$ will share high level traits from both v and w, therefore making it a possible representation extracted from a video depicting similar actions and entities as them.

3.3.2 Textual side of the proposed multimodal augmentation

As in the case of videos, we design the textual augmentation technique in the feature space. We define two criteria, $\psi_V(a, q)$ and $\psi_N(o, q)$, to identify the captions which can become valid substitutes of a given q based on one of its actions a or entities o. For instance, $\psi_V(a, q) = \{d \,|\, a \in \texttt{act}(d) \wedge \texttt{ent}(q) \cap \texttt{ent}(d) \neq \emptyset\}$.

Given these operators and a caption q, the augmentation is performed with chance χ, and the decision between actions and entities is taken with uniform chance (χ is the same as in Section 3.3.1). After the selection of a valid candidate d (step 16), the latent representations of both q and d are extracted with a function g (steps 3 and 18) and then mixed with the function ρ (step 19). As for the videos, we define ρ as a mixing function working on the high level concepts extracted from the language model g, that is $\rho(\overline{q}, \overline{d}) = \lambda \cdot \overline{q} + (1 - \lambda) \cdot \overline{d}$ ('Text fusion' in Fig.3.1).

3.4 Experimental results

To empirically validate our methodology, we present several experiments performed on two public datasets: YouCook2 [343], a popular dataset of around 13000 video clips on complex kitchen activities, and the recently released EPIC-Kitchens-100 [45], a challenging and large scale public dataset comprising more than 70000 egocentric video clips, *i.e.* the videos are taken from a first-person perspective by leveraging wearable cameras. The videos capture multiple daily activities in a kitchen and the camera wearers do not follow any scripted interaction. Each video is annotated with a caption, which is provided by a human annotator and contains at least one verb and one or more nouns. Additionally, verbs and nouns are respectively grouped into 98 and 300 semantic classes, each of which contains semantically close tokens, *e.g.* the class for verb 'take' also contains 'pick up', 'grab', *etc.* An example of these data is shown in Figure 3.2. Given the multimodal nature of the videos, we use the RGB, flow, and audio features extracted with TBN [134], which are provided alongside the dataset. When dealing with YouCook2, we use the features extracted with S3D pretrained on HowTo100M [195, 193] which are available within the VALUE benchmark [158].

In the context of the EPIC-Kitchens-100 multi-instance retrieval challenge[1], Damen et al. use two rank-aware metrics, the Mean Average Precision (mAP) [10] and the Normalized Discounted Cumulative Gain (nDCG) [122] to report performance. Both

[1] https://epic-kitchens.github.io/2022#challenge-action-retrieval

caption: pour washing up liquid into sponge
verb: pour (id=9)
nouns: washing up liquid (id=22), sponge (id=9)

caption: open sausage box
verb: open (id=3)
nouns: sausage box (id=23)

Figure 3.2: Examples of the data used in EPIC-Kitchens-100. Verbs and nouns are grouped into semantic classes containing tokens which share similar semantics, *e.g.* class 22 for nouns contains 'washing up liquid', but also 'cleaning liquid', 'detergent', *etc.*

are defined in terms of the following relevance function [45]:

$$\mathcal{R}(x,y) = \frac{1}{2}\Big(\frac{|x^V \cap y^V|}{|x^V \cup y^V|} + \frac{|x^N \cap y^N|}{|x^N \cup y^N|}\Big)$$

where x^N, x^V, y^N, and y^V are sets of noun and verb semantic classes observed in captions x and y. When x or y are videos, the associated caption is considered. The mAP uses a binary definition of relevance, meaning that either a caption (or video) is relevant to the query, *i.e.* the computed relevance is 1, or it is not. The nDCG uses a finer-grained definition of relevance, allowing continuous values between 0 and 1.

To validate the proposed data augmentation technique, we use a text-video retrieval model to perform the alignment between the visual and textual features. In particular, we chose HGR [27] as the baseline because of its proven capabilities on multiple datasets, including EPIC-Kitchens-100 [70]. To compute the descriptors of the input data, HGR builds a graph structure of the caption and aggregates it with a graph neural network, whereas it relies on simpler neural networks for the video. We follow their hyperparameters setting and perform the training for 50 epochs on EPIC-Kitchens-100 and for 125 epochs on YouCook2, in both cases with a batch size of 64. We release code and pretrained models on Github at https://github.com/aranciokov/FSMMDA_VideoRetrieval.

3.4.1 Visual augmentation

We start by exploring the effectiveness of our video augmentation technique. First of all, in Sections 3.4.1 and 3.4.1 we perform ablation studies on two 'parameters' of our

strategy, which are the granularity of the selection criteria and the influence of the λ parameter. Then, in Section 3.4.1 we compare our technique to another technique from the literature.

Video selection criteria

In our video augmentation technique, we define two fine-grained criteria to identify which videos are valid candidates, *i.e.* sharing similar semantics, for the augmentation of a given v (see Section 3.3.1). The criteria are defined for both actions and entities, and identify all the training videos which share a specified class (*e.g.* the action 'take') and at least one semantic class of the other type (*e.g.* the entity 'slice of pizza'). Here we explore a coarser definition of the criteria, by only guaranteeing that the specified class (*e.g.* 'take') is shared. As an example, given a video v and the action 'take', the fine-grained criterion selects videos which depict an action from the same class as 'take' and at least one of the entities shown in v; the coarser criterion ignores the latter constraint, therefore identifying many more videos as viable candidates.

We depict the results of this inquiry in Figure 3.3 with the orange ('coarse, $\lambda \sim \beta(1,1)$') and red ('fine, $\lambda \sim \beta(1,1)$') curves. We also show the values obtained by the HGR baseline, which does not perform the augmentation, with a blue dashed line. Considering that for each sample the augmentation happens with chance χ (see Alg. 1, steps 4-5), we vary $\chi \in \{25\%, 50\%, 75\%, 100\%\}$. As defined in our method, we sample the λ parameter of the mixing function (see Section 3.3.1) from a Beta distribution with both parameters set to 1. Both with the fine-grained and the coarse criterion, we observe that the nDCG on the test set increases as the augmentation is performed more frequently: the fine-grained criterion leads to 37.8% average nDCG when $\chi = 25\%$ and up to 40.9% when the augmentation is always done ($\chi = 100\%$), whereas the coarser criterion leads to nDCG values ranging from 38.6% ($\chi = 25\%$) to 41.8% ($\chi = 100\%$). The difference in nDCG is likely explained by the weaker constraint employed by the coarse criterion to identify the videos used for the augmentation: since the candidates are only required to share one of the semantic classes of the original video, the augmented training samples likely cover a wider set of high level concepts. This helps the trained model retrieving partially (and minimally) relevant videos and captions at inference time. However, the fine-grained criterion leads to higher quality ranked lists as confirmed by the mAP (45.6% compared to less than 42% obtained by the coarse criterion), which suggests that the highly relevant captions and videos are retrieved at the top ranks. While the sum of recalls (Rsum) shows higher values for the coarse criterion, it is not as relevant as the other metrics: in fact, the recall solely keeps track of the 'groundtruth' associations, but many captions may equally describe the same video and this can not be captured through the recall. As an example, if q_1 = "pick a slice of pizza" and q_2 = "grab a slice of pizza" were the first retrieved captions for a video originally paired with q_2, mAP and nDCG would be invariant with respect to the ordering, whereas the recall metrics would not (*e.g.* R@1 would be 0 in this case).

Influence of the mixing parameter λ on the final performance

The main parameter of the mixing function we use is λ, which represents the extent to which the original video features are mixed with the features from a different video

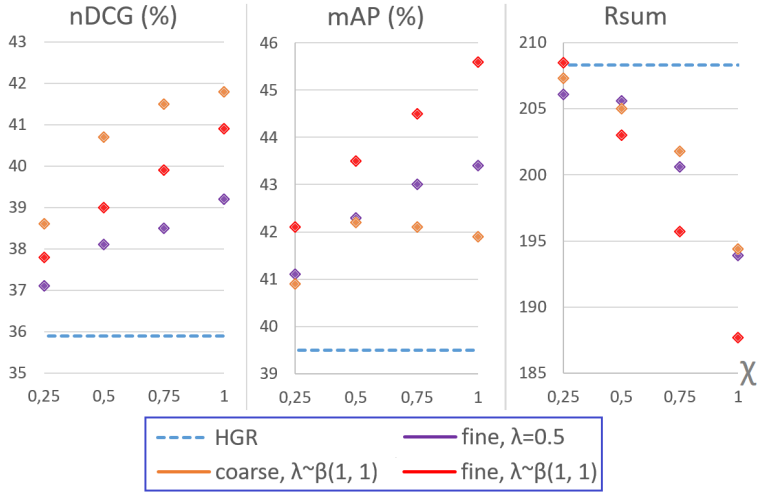


Figure 3.3: Video augmentation. (blue) Performance of the baseline without augmentation. (red) The proposed video augmentation technique (see Sec. 3.3.1). (orange) We explore a coarser selection criterion (see Sec. 3.4.1) to identify the videos used to perform the mixing. (purple) We explore a fixed solution (see Sec. 3.4.1) for the λ parameter of our mixing function. Performance is displayed as the parameter χ, used to determine how frequently the augmentation is performed, varies from 0 (0%) to 1 (100%).

(see Sec. 3.3.1 for more details). Therefore, as a second experiment we explore a fixed solution for λ in place of the variable solution defined in our method. In particular, we experiment with $\lambda = 0.5$, which is the expected value of λ under the Beta distribution. As before, we analyze the performance as χ varies, and depict the results in Figure 3.3 with the red ('fine, $\lambda \sim \beta(1, 1)$') and purple ('fine, $\lambda = 0.5$') curves.

If we compare the two variants of λ, three observations can be made. First of all, as in the previous case, we observe that also with $\lambda = 0.5$ the performance improves as the augmentation becomes more frequent: in fact, when compared to the non-augmented baseline (35.9% average nDCG and 39.5% average mAP, depicted with the blue dashed line), we observe better nDCG and mAP rates, leading to up to 39.2% nDCG and 43.4% mAP when the video is always replaced with its augmented version. Secondly, in both cases the best performance are achieved when the video is always ($\chi = 100\%$) replaced with its augmented version. Thirdly, a variable λ is preferred: in fact, the usage of a variable λ consistently leads to an improvement in both nDCG (+1.7%) and mAP (+2.2%).

Comparison with other visual augmentation techniques

As mentioned before, we compare our proposed video augmentation technique to the only other solution working in the feature space, that is the video-level augmentation proposed by Dong et al. [56, 58], and use the code publicly shared by the authors. We illustrate the results in Figure 3.4, where we plot the baseline in blue, our proposed

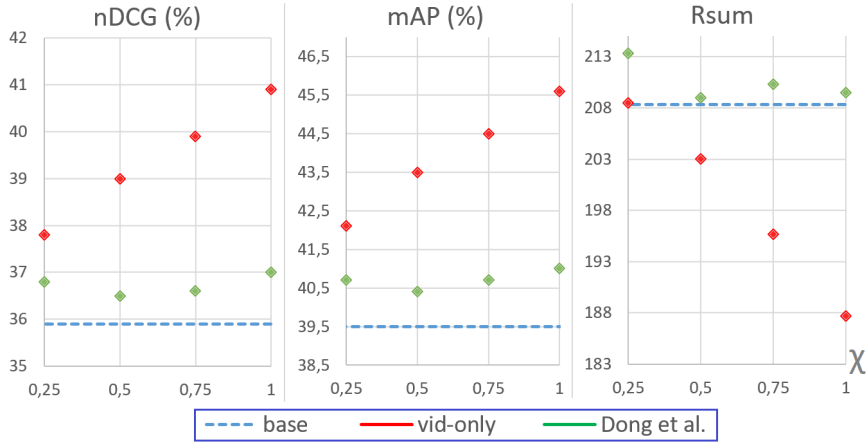


Figure 3.4: Video augmentation. (red) Our proposed video augmentation technique. (green) We adapt the video-level augmentation by Dong et al. [56, 58] in our framework. With our technique, we achieve much higher nDCG and mAP, therefore retrieving more semantically similar captions and videos at the top of the ranked list.

video augmentation technique in red, and the augmentation by Dong et al. in green. A better Rsum is observed with the latter, meaning that the groundtruth is more likely to be retrieved at the top of the ranked list, but this metric ignores that other captions and videos may have the same semantics. On the other hand, it can be seen that our technique let us achieve higher quality ranked lists with a margin of more than 4% both in nDCG and mAP.

3.4.2 Textual augmentation

Before diving into the joint augmentation of video and text, we explore the effects of text augmentation on retrieval performance. We start by exploring how the performance are affected based on how frequently the augmentation happens, so we vary $\chi \in \{25\%, 50\%, 75\%, 100\%\}$ and display the results in Figure 3.5 with the grey curve, whereas the value obtained without any augmentation is shown with the blue line. As in the previous case the proposed augmentation is greatly useful, leading to improvements of up to +4.5% nDCG (40.4% compared to 35.9% obtained by the baseline) and +6.2% mAP (45.7% compared to 39.5%) when the augmentation is always performed.

Then, we perform a comparison with a symbolic technique inspired by the works of Wei et al. and Wang et al. [294, 287], which consists in replacing a word with a synonym. Although it works on the raw textual data, we chose this technique because performing the synonym replacement shares some similarities with how we select the candidate for the mixing step. We report the results in Figure 3.5 with the orange curve. Two major observations can be made. First of all, the performance increases with χ, as in the previous cases, although it reaches a peak in the mAP performance when $\chi = 75\%$. Secondly, it leads to an improvement over the baseline, but the proposed technique achieves better performance obtaining a margin of +2.1% nDCG (40.4% compared to

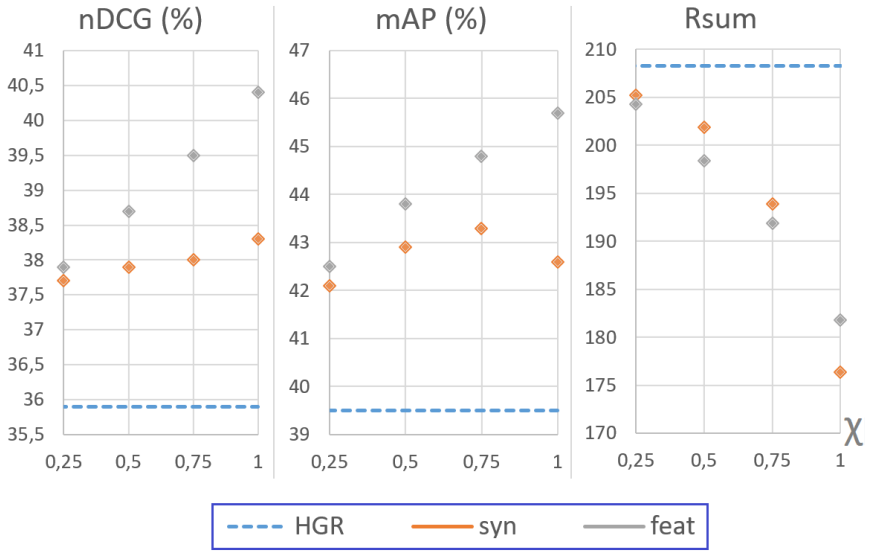


Figure 3.5: Text augmentation. Experiments on EPIC-Kitchens-100. (grey) The proposed method which performs the augmentation in the feature space. (orange) New captions are created by performing synonym replacement (see Sec. 3.4.2 for details). We observe consistent improvements over the baseline in both cases, but the proposed feature-space augmentation leads to overall better results.

38.3%) and +2.4% mAP (45.7% to 43.3%).

3.4.3 Joint text-video augmentation

In the previous experiments we show that the two components of the proposed multimodal data augmentation technique are greatly useful and improve the performance on unseen test examples. To show the usefulness of our complete technique, we compare its performance to the two unimodal components. In Figure 3.6 we display how the final performance varies with the parameter χ. We observe two major results. First of all, if only one of the two unimodal components is used (video-only in orange, text-only in green), then we observe higher nDCG when the video is augmented, and slightly higher mAP when the captions are augmented. Secondly, considerable improvements are achieved when the complete multimodal technique is adopted during training, leading to a margin of more than 1% on both metrics.

3.4.4 Synergy with improved selection of contrastive samples

To validate the robustness of our data augmentation strategy, we test it on two recently published techniques: RAN and RANP [70]. RAN and RANP are two online mining techniques introduced for a contrastive framework which lead to increased text-video retrieval performance by improving the selection of both negative and positive examples. As done in the previous experiments, we explore how these techniques affect our

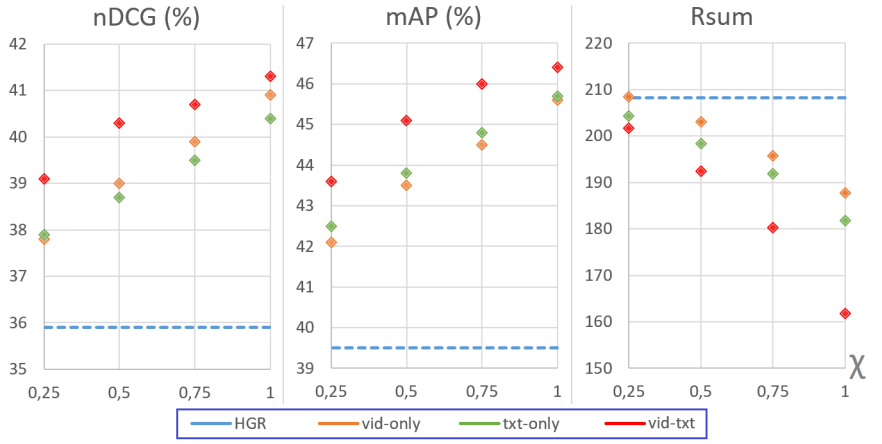


Figure 3.6: Comparison between the baseline (blue), our proposed multimodal technique (red), and its two components, video-only (orange) and text-only (green). Experiments on EPIC-Kitchens-100.

framework while varying χ and visualize the results in Figure 3.7, where HGR is shown in blue, RAN and RANP with light and dark green, our proposed multimodal technique with orange, and the addition of RAN and RANP to our method is depicted with dark orange and red. We observe that our proposed technique and the improved selection of negative examples provided by RAN synergize well: in fact, with this addition we obtain up to +12% nDCG and +2.2% mAP, which also leads to a margin of 4.3% nDCG and 1.7% mAP over RAN, as shown by the dark orange curve and light green dashed line in Fig. 3.7. Conversely, the addition of RANP, which adds positive examples mining to the contrastive loss, leads our method to similar nDCG rates but worse mAP when the augmentation is always performed ($\chi = 100\%$), therefore we observe a lesser synergy between the two. Finally, in Table 3.1 we report a quantitative comparison between augmented and non-augmented versions of HGR, RAN, and RANP. For the non-augmented versions, we report the same results observed in [70]. For the augmented HGR, RAN, and RANP we report nDCG and mAP observed with χ respectively set to 100%, 75%, and 50% (selected by looking at Fig. 3.7). It can be seen that in almost all the cases, both looking at text-to-video ('t2v'), video-to-text ('v2t'), and text-video retrieval ('t-v'), further improvements can be obtained by using the proposed augmentation technique.

3.4.5 Comparison to state-of-the-art

In Table 3.2 we compare our results to all the published methods for the EPIC-Kitchens-100 dataset, including the baseline we used, MME and JPoSE by Wray et al. [299], Hao et al. from the technical report of last year challenge [46], and RANP by Falcon et al. [70]. As can be seen, by leveraging our proposed multimodal data augmentation technique on the state-of-the-art methods RAN and RANP, we achieve further improvements.

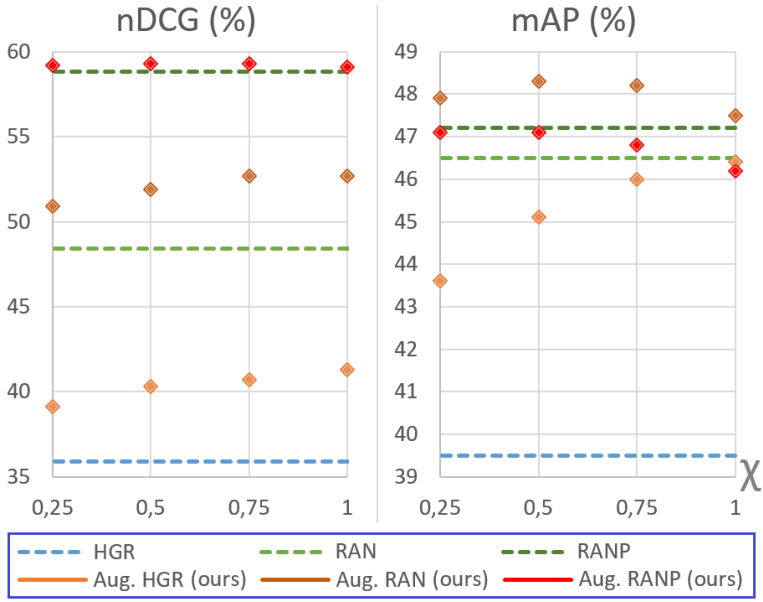


Figure 3.7: Comparison with RAN and RANP [70]. Three non-augmented methods: (blue) HGR; (light green) RAN; (dark green) RANP. The three methods are then augmented with our proposed multimodal augmentation technique, leading to improved results: (orange) augmented HGR; (dark orange) augmented RAN; (red) augmented RANP. Best viewed in color.

Table 3.1: Comparison between HGR, RAN, and RANP and the three methods augmented with our proposed multimodal data augmentation technique. We observe that our technique synergizes well with different techniques, leading to improved performance both in terms of mAP and nDCG.

	nDCG (%)			mAP (%)		
Model	t2v	v2t	t-v	t2v	v2t	t-v
HGR [27]	37.9	41.2	39.5	35.7	36.1	35.9
Aug. HGR (ours)	**41.0**	**41.6**	**41.3**	**42.6**	**50.2**	**46.4**
RAN [70]	47.1	49.7	48.4	43.1	49.9	46.5
Aug. RAN (ours)	**51.6**	**53.8**	**52.7**	**44.1**	**52.4**	**48.2**
RANP [70]	56.5	61.2	58.8	**42.3**	52.0	**47.2**
Aug. RANP (ours)	**57.2**	**61.4**	**59.3**	41.9	**52.4**	**47.2**

Table 3.2: Comparison with the baseline and state-of-the-art methods for EPIC-Kitchens-100 (results for MME and JPoSE are from [45], Hao et al. from [46]). With the proposed multimodal data augmentation technique, we observe higher mAP performance, therefore more highly relevant captions and videos are retrieved at the top ranks, when compared to other techniques.

	EPIC-Kitchens-100					
	nDCG (%)			mAP (%)		
Model	t2v	v2t	t-v	t2v	v2t	t-v
HGR [27]	37.9	41.2	39.5	35.7	36.1	35.9
MME [299]	46.9	50.0	48.5	34.0	43.0	38.5
JPoSE [299]	51.5	55.5	53.5	38.1	49.9	44.0
Hao et al. [46]	51.8	55.3	53.5	38.5	50.0	44.2
RANP [70]	56.5	61.2	58.8	42.3	52.0	47.2
Aug. RAN (ours)	51.6	53.8	52.7	**44.1**	**52.4**	**48.2**
Aug. RANP (ours)	**57.2**	**61.4**	**59.3**	41.9	**52.4**	47.1

Table 3.3: Comparison with the HGR baseline on YouCook2. The augmented version uses the proposed multimodal data augmentation technique with $\chi = 0.50$.

	YouCook2					
	nDCG (%)			mAP (%)		
Backbone & Model	t2v	v2t	t-v	t2v	v2t	t-v
S3D HGR [27]	50.1	49.7	49.9	45.3	**43.9**	44.6
S3D Aug. HGR (ours)	**50.8**	**51.3**	**51.0**	**45.4**	**43.9**	**44.7**

Moreover, in Table 3.3 we show that the proposed technique also leads to improvements on YouCook2. For this dataset, we use publicly available features (from the VALUE benchmark [158]) which were extracted with an HowTo100M-pretrained S3D model [193]. In particular, by using the proposed technique with $\chi = 0.50$, we observe +1.1% nDCG on average, reaching 51.0% nDCG. On the other hand, lesser improvements are observed in terms of mAP.

3.5 Conclusions

In this Chapter, we introduced a multimodal data augmentation technique working in the feature space. In this way several advantages can be leveraged, including the possibility to work on the high level concepts extracted from the deeper layers of CNN-based backbones and easier applicability since the original videos need not to be shared, avoiding copyright and privacy issues. To validate our solution, we performed multiple experiments on the large scale public dataset EPIC-Kitchens-100, as well as a comparison on YouCook2. We tested our technique on three different methods, including recent state-of-the-art methods on EPIC-Kitchens-100, and achieved further improvements. As a future work, we plan to extend our technique to different datasets (*e.g.* MSR-VTT [310] and VATEX [289]) and methods (*e.g.* dual encoding by [57]).

4

Text-to-Image Synthesis Based on Machine Generated Captions

4.1 Introduction

Text-to-Image Synthesis, also called Conditional Image Generation, is a process that consists in generating a photo-realistic image given a textual description. It is a challenging task and it is revolutionizing many real-world applications. For example, starting from a Digital Library of adventure books it could be possible to enrich the reading experience with computer-generated images of the locations explored in the story, while a Digital Library of recipe books may be enriched with images representing the steps involved in a given recipe. In addition, such images may be used to exploit Information Retrieval systems based on visual similarity. Due to its great potentiality and usefulness, it raised a lot of interest in the research fields of Computer Vision, Natural Language Processing, and Digital Libraries.

One of the main approaches used for the text-to-image task involves the use of Generative Adversarial Networks (GAN) [85]: starting from a given textual description, GANs can be conditioned on text [231], [230], [332] in order generate high-quality images that are highly related to the text meaning.

To condition a GAN on text, captioned images datasets are needed, meaning that one (or more) captions must be associated to each image. Despite the large amount of uncaptioned images datasets, the number of captioned datasets is limited. For example, LSUN [320] dataset, which consists in more than 59 million labeled images for each of 10 scene categories and 20 object categories [320]. The LSUN-bedroom dataset contains images from LSUN dataset tagged with the "bedroom" scene category. It contains around $\sim 3,000,000$ images [320], but it does not contain the associated captions. This may lead to a difficulty in training a conditional GAN to generate bedroom images related to a given textual description, such as "a bedroom with blue walls, white furniture and a large bed". In this Chapter we propose an innovative, though quite simple approach to address this issue as shown in Figure 4.1. First of all, a captioning system (that we

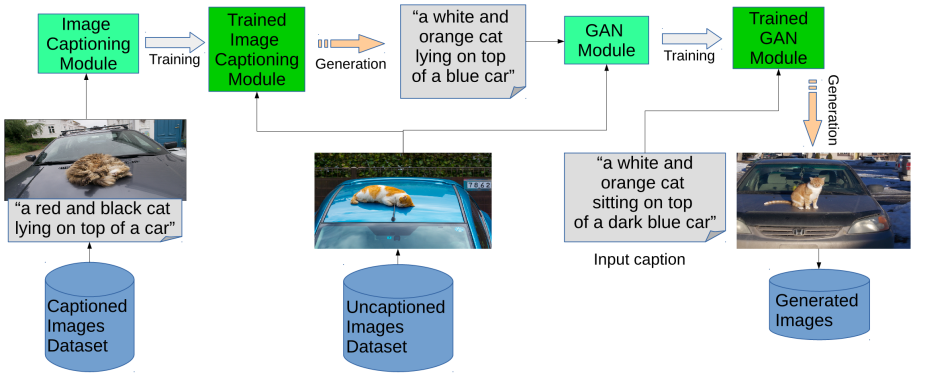


Figure 4.1: Our pipeline: captioned images are used to train the Image Captioning Module; uncaptioned images are then captioned through the Trained Image Captioning Module and both the image and the generated captions are used to train the GAN Module; finally, the Trained GAN Module is used to generate an image based on an input caption.

call Image Captioning Module) is trained on a generic captioned dataset and used to generate a caption for the uncaptioned images. Then, the conditional GAN (that we call GAN Module) is trained on both the input image and the "machine-generated" caption. A high-level representation of the architecture is shown in Figure 4.2. To evaluate the results, the performance of the GAN using "machine-generated" captions are compared with the results obtained by the unconditional GAN. To test and evaluate our pipeline, we are using the LSUN-bedroom [320] dataset.

The results obtained in the experiments are very preliminary yet very promising. According to the results observed in this Chapter, the GAN Module does not learn how to produce meaningful images, with respect to the caption meaning, and we hypothesize that this is due to the "machine-generated" captions we use to condition the GAN Module. The Image Captioning module is trained on the COCO dataset [170], which contains captioned images for many different classes of objects and intuitively this should lead the Image Captioning Module to learn how to produce captions for bedroom images as well. Despite being able to produce the desired captions, we notice that the "machine-generated" captions are often too similar and not detailed for different bedroom images. The last section of this Chapter proposes some approaches that can deal with these problems.

4.2 Related Work

In 2014, Goodfellow et al. introduced Generative Adversarial Networks (GAN) [85], a generative model framework that consists in training simultaneously two models: a generator network and a discriminator one. The generator network has the task of generating images as real as possible, while the discriminator network has to distinguish the generated images from the real ones. Generative models are trained to implicitly

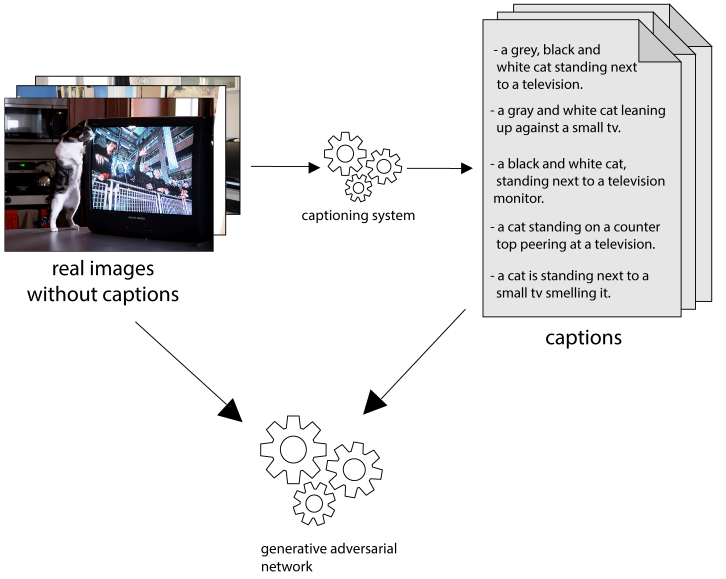


Figure 4.2: Pipeline: images are fed to a captioning system that outputs its captions. The generated captions and the images are then given as input for training the conditional GAN.

capture the statistical distribution of training data; once trained, they can synthesize novel data samples, which can be used for example in the tasks of semantic image editing [346] and data augmentation [18].

GANs can be trained to sample from a given data distribution, in such case a random vector is provided as input to the generator. Otherwise, as in the case of text-to-image synthesis, they can be trained conditionally, meaning that an additional variable is provided as input to control the generator output. In certain formulations, the discriminator observes the conditioning variable too, during training. In the literature, several possibilities were tested for the variables used to condition a GAN: attributes or class labels (e.g. [30], [204]), images (e.g. for the tasks of photo editing [346] and domain transfer [118]).

Several methods have been developed to generate images conditioned on text. Mansimov et al. [188] built an AlignDRAW model trained to learn the correspondence between text and generated images. Reed et al. in [232] used PixelCNN to generate images using both text descriptions and object location constraints. Nguyen et al. [203] used an approximate Langevin sampling approach to generate images conditioned on text, but it required an inefficient iterative optimization process. In [231], Reed et al. successfully generated 64×64 images for birds and flowers conditioning on text descriptions. In their follow-up work [230], they were able to generate 128×128 images by using additional annotations on object part locations. Denton et al. in [51] proposed the Laplacian pyramid framework (LAPGANs), which is composed of a series of GANs. A residual image is conditioned at each level of the pyramid on the image of the previous stage to produce an image for the next stage. Also in [133], Kerras et al. use

a similar approach by incrementally adding more layers in the generator and in the discriminator. [331] and [332] suggest the use of a so-called sketch-refinement process, where the images are first generated at low resolutions using a GAN conditioned over the textual description, and then refined with another GAN conditioned on both the image generated at the previous step and the input textual description. [105] and [161] infer a semantic label map by predicting bounding boxes and object shapes from the text, and then synthesize an image conditioned on the layout and the text description. A recent work by Qiao et al. [222] uses a three-step approach where it first computes word- and sentence-level embedding from the given textual description, then it uses the embeddings to generate images in a cascaded architecture, and finally starting from the image generated at the previous step it tries to regenerate the original textual description, in order to semantically align with it. Although several different state-of-the-art architectures may be chosen for the task, such as HDGAN [338] and AttGAN [311], in our pipeline we decided to use StackGAN-v2 [332] as the conditional GAN component, given the availability of its code on GitHub.

Recently, several impressive results [344], [167], [234] were obtained for the Image Captioning (or image-to-text) task, which deals with the generation of a caption describing the given image and the objects taking part to it. It is an important task that raises a lot of interest in the Computer Vision and Natural Language Processing research fields. A recent and comprehensive survey about the task is provided by Hossain et al. in [107]. Some of the approaches used for this task involve the use of Encoder/Decoder networks and Reinforcement learning techniques.

The encoder/decoder paradigm for machine translation using recurrent neural networks (RNNs) [35] inspired [131], [280] to use a deep convolutional neural network to encode the input image, and a Long Short-Term Memory (LSTM) [104] RNN decoder to generate the output caption. Given the unavailability of labeled data, recent approaches to the image captioning task involve the use of reinforcement learning and unsupervised learning-based techniques. [344] and [167] use actor-critic reinforcement learning methods, where a "policy network" (the actor) is trained to predict the next word based on the current state, whereas a "value network" (the critic) is trained to estimate the reward of each generated word. These techniques overcome the need to sample from the policy (actors) action space, which can be enormous, at the expense of estimating future rewards. Another approach, used by Ranzato et al. in [226], consists in applying the REINFORCE algorithm [295]. A limitation of this algorithm consists in the requirement of a context-dependent normalization to tackle the high variance encountered when using mini-batches. The approach we are following uses Self-Critical Sequence Training (SCST) [234] which is a REINFORCE algorithm that utilizes the output of its own test-time inference algorithm to normalize the rewards it experiences: doing so, it does not need neither to estimate the reward signal nor the normalization.

4.3 Our Approach

We propose a pipeline whose goal is to generate images by conditioning on "machine-generated" captions. This is fundamental when image captions are not available for a specific domain of interest. Thus, the proposed solution involves the use of a generic captioned dataset, such as the COCO dataset, to make the Image Captioning Module

capable of generating captions for a specific domain.

To do so, we want to explore the possibility of using an automatic system to generate textual captions for the images and use them for the training of a Generative Adversarial Network. For achieving our goal, we built a pipeline composed by an Image Captioning Module and a GAN Module, as shown in Figure 4.1. First of all, the Image Captioning Module is trained over a generic captioned dataset to generate multiple captions for the input image. Then, real images are given as input to the Trained Image Captioning Module, which outputs multiple captions for each image. The generated captions together with the images are then fed to the GAN Module, which learns to generate images conditioned on the "machine-generated" captions. By feeding the GAN with multiple captions for each image, the GAN can better learn the correspondence between images and captions.

In the following sections, we detail the two modules used in our pipeline: the Image Captioning Module and the GAN Module.

4.3.1 Image Captioning Module

The goal of the Image Captioning Module is to generate a natural language description of an image. Good performance in this task are obtained by learning a model which is able to first understand the scene described in the image, the objects taking part to it and the relationships between them, and then to compose a natural language sentence describing the whole picture. Given the complexity of such a task, it is still an open challenge in the fields of Natural Language Processing and Computer Vision. The task of open domain captioning is a challenging task. It requires a fine-grained understanding of the whole entities, attributes and relationships in an image. In our pipeline, we are implementing our Image Captioning Module in a similar way as the one proposed in [234], meaning that we also use a captioning system based on FC models. It has been built using an optimization approach that is called Self-Critical Sequence Training (SCST).

Typical deep learning models used for the Image Captioning task are trained with the "teacher-forcing" technique, which consists in maximizing the likelihood of the next ground-truth word given the previous ground-truth word. This has been shown to generate some mismatches between the training and the inference phase, knows as "exposure bias". Moreover, the metrics used during the testing phase are non-differentiable (such as BLEU and CIDEr), meaning that the captioning model can not be trained to directly optimize them. To overcome these problems, Reinforcement Learning techniques such as the REINFORCE algorithm have been used. SCST is a variation and an improvement of the popular REINFORCE algorithm that, rather than estimating a baseline to normalize the rewards and reduce variance, utilizes the output of its own test-time inference algorithm to normalize the rewards it experiences. This means that it is forced to improve the performance of the model under the inference algorithm used at test time. Practically, SCST has much lower variance than REINFORCE and can be more effectively trained on mini-batches of samples using SGD. Moreover, it has been shown that SCST system has achieved state-of-the-art performance by optimizing their system using the test metrics of the MSCOCO task. Practically, it has been found that SCST has much lower variance, and can be more effectively trained on mini-batches of samples using SGD. Since the SCST baseline is based on the test-time estimate under

the current model, SCST is forced to improve the performance of the model under the inference algorithm used at test time. In addition, this encourages training consistency like the maximum likelihood-based approaches except it optimized sequence metrics.

4.3.2 GAN Module

The GAN Module has the major role of learning to generate images by conditioning on the "machine-generated" captions. In particular, we are using StackGAN-v2 [332] as our GAN Module.

StackGAN-v2 consists of a multiple stage generation process, where high-resolution images are obtained by initially generating low-resolution images which are then refined in multiple steps. It consists in a single end-to-end network composed by multiple generators and discriminators in a tree-like structure. Different branches of the tree generate images of different resolutions: at branch i, the generator G_i learns the image distribution p_{G_i} at that scale, while the discriminator D_i estimates the probability of a sample being real. The framework of StackGAN-v2 has a tree-like structure, that takes as input the noise vector $z \sim p_{noise}$. The noise z is transformed in hidden feature layer by layer. The hidden features h_i for each generator G_i are calculated by a non-linear transformation

$$h_0 = F_0(z); \quad h_i = F_i(h_{i-1}, z), \tag{4.1}$$

where h_i represents hidden features for the i^{th} branch, m is the total number of branches, and F_i are modeled as neural networks. The noise vector z is concatenated to the hidden features h_{i-1} as the inputs of F_i for calculating h_i. The generators produce samples at different scales $(s_0, s1, ..., s_{m-1})$ based on the hidden features at different layers $(h_0, h_1, ..., h_{m-1})$.

$$s_i = G_i(h_i), \quad i = 0, 1, ..., m-1, \tag{4.2}$$

where G_i is the generator for the i^{th} branch. Since we are more interested in the conditional case, we are not reporting the loss function used by the generator and the discriminator in the unconditional setting, for which more details can be found in [332]. The discriminator D_i takes a real image x_i or a fake sample s_i as input and is trained to classify them as real or fake by minimizing the cross entropy loss:

$$\begin{aligned} \mathcal{L}_{D_i} = & \underbrace{-\mathbb{E}_{x_i \sim p_{data_i}}[logD_i(x_i)] - \mathbb{E}_{x_i \sim p_{G_i}}[log(1 - D_i(s_i))]}_{\text{unconditional loss}} \\ & \underbrace{-\mathbb{E}_{x_i \sim p_{data_i}}[logD_i(x_i, c)] - \mathbb{E}_{x_i \sim p_{G_i}}[log(1 - D_i(s_i, c))]}_{\text{conditional loss}} \end{aligned} \tag{4.3}$$

where x_i is an image from the true image distribution p_{data_i} at the i^{th} scale, s_i is from the model distribution p_{G_i} at the same scale. While StackGAN-v2 [332] follows the approach of Reed et al. [229] to pre-train a text encoder to extract visually-discriminative text embeddings of the given description, in our case we use Skip-Thought [143], that works at the sentence level, to generate the text embeddings (c in the equations 4.3 and 4.4). Sentences that share semantic and syntactic properties are mapped to corresponding vector representations [143].

The multiple discriminators are trained in parallel each one for a different scale, while the generator is instead optimized to jointly approximate multi-scale image distributions $p_{data_0}, p_{data_1}, ..., p_{data_{m-1}}$ by minimizing the following loss function:

$$\mathcal{L}_G = \sum_{i=1}^{m} \mathcal{L}_{G_i}, \quad \mathcal{L}_{G_i} = \underbrace{-\mathbb{E}_{s_i \sim p_{G_i}}[logD_i(s_i)]}_{\text{unconditional loss}} \underbrace{-\mathbb{E}_{s_i \sim p_{G_i}}[logD_i(s_i, c)]}_{\text{conditional loss}} \tag{4.4}$$

where L_{G_i} is the loss function for approximating the image distribution at the i^{th} scale. The unconditional loss is used to determine whether the image is real or fake, while the conditional loss is used to determine if the image and the condition match.

4.4 Experimental Results

In this section, we present the preliminary results of the experiments involving the proposed pipeline. The Image Captioning Module was trained on the COCO dataset [170], which contains 120,000 generic images tagged with categories and captioned by five different sentences each. The uncaptioned dataset that we considered is the LSUN [320] dataset, which consists in more than 59 million labeled images. From the LSUN dataset, we first select the $\sim 3,000,000$ images tagged with the "bedroom" scene category and from that set a subset of the first 120,000 images is selected: 80,000 are then used to train the GAN and 40,000 as test set. Later on in this Chapter, the selection of the $\sim 3,000,000$ images tagged with the "bedroom" scene category is called "LSUN-bedroom".

A typical metric used to evaluate both the quality and the diversity of generated images is the Inception Score [240]. Unfortunately, the type of image of the LSUN dataset is very different from those used by ImageNet [332, 50], therefore it has been shown that the Inception Score is not a good indicator for the quality of generated images [332]. So we decided not to report the obtained scores.

We performed three experiments over the considered dataset.

The first experiment consists in training the GAN Module on the whole LSUN-bedroom dataset ($\sim 3,000,000$ images). This is done because of two reasons: first, it serves as a baseline for the next experiment; second, we compare the results obtained by our computing facilities with the results obtained in [332], since with our graphics card we are limited to a lower batch size of 16. Figure 4.4 shows some examples of generated images, and it is possible to see that the quality of the generated images is similar to those shown in Figure 4.3 [332].

To reproduce the results reported in the paper, we used an NVIDIA GTX 1080 8GB machine. It took us around one month to train the GAN Module on the whole LSUN dataset. Because of this, we decided to explore and understand how the GAN Module performs with less training images. In the second experiment, the training of the GAN Module without conditioning is done on a subset of LSUN-bedroom, consisting of 120,000 images. Some of the results obtained in this experiment are showed in Figure 4.5. Although the quality of the generated images is slightly reduced, it is possible to see that the semantic content is still clear and defined.

Finally, to test our pipeline, we used the Image Captioning Module to generate cap-

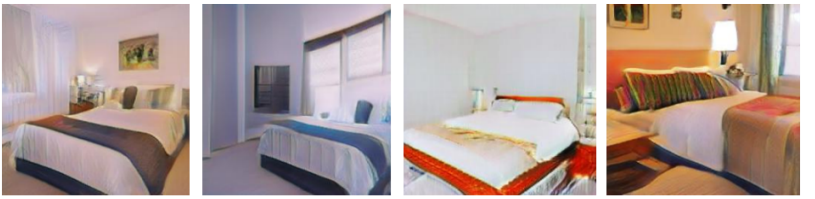


Figure 4.3: Examples of images generated by the StackGAN Module trained on the whole LSUN-bedroom dataset.

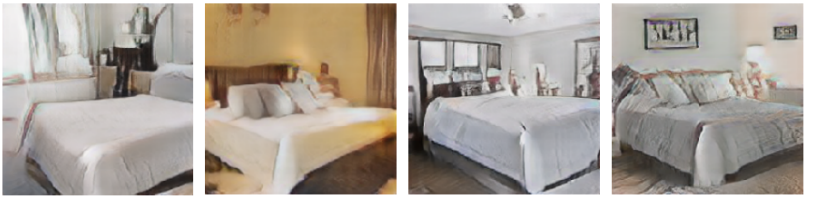


Figure 4.4: Examples of images generated by the GAN Module trained on the whole LSUN-bedroom dataset.

tions for the images contained in the considered subset of the LSUN-bedroom dataset. Then, the GAN Module was trained on these same images and conditioned by the "machine-generated" captions. About the preliminary results that we obtained, some examples are shown in Figure 4.6. We suspect the problem is due to the similarity of the "machine-generated" captions: the LSUN-bedroom dataset does not come with captions and thus the Image Captioning Module is trained on a generic dataset (COCO) and not for that specific dataset. Because of this, the Image Captioning Module is unable to produce detailed and varied captions for different bedroom images. For instance, Table 4.1 reports some examples of generated captions on images taken from LSUN-bedroom, which show that the Image Captioning Module is often unable to generate detailed and accurate captions which compete with those provided by human annotators. In addition, usually GANs used noise vector to generate images which always different from each other [85]. In our experiment, the noise vector is taken as input by the model and used to generate an image. Then, the captions are used to yield the embeddings, which are also used as noise by the generator. The fact that the noise is almost always the same could be the cause of the observed problem.

We found that the scores for the LSUN-bedroom dataset seem to not fully correlate with the quality of the generated images. As explained in [332], this may be due to the inception score being trained on the inception dataset, and thus it does not work well on datasets with specific types of images. Also, it has to be considered that different datasets get inception scores in different ranges. For this reason, inception scores must not be compared across different datasets.

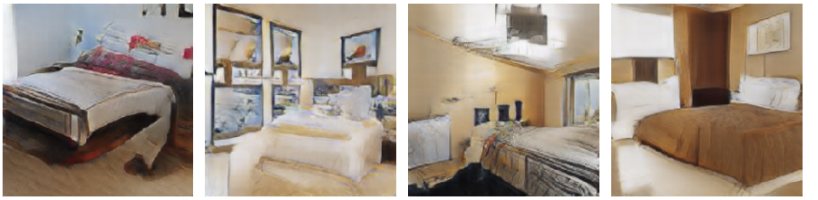
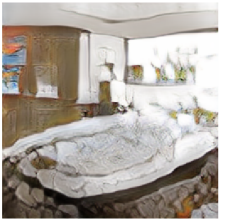
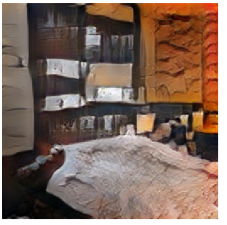


Figure 4.5: Examples of images generated by the GAN Module trained on a part of the LSUN-bedroom dataset.

(a) “this bedroom looks very old-fashioned”



(b) “the bedroom has carpeted floor and the walls are papered”



(c) “the room has rich colors and overall looks very modern”

(d) “this small bedroom has matching wooden and glass furniture”

Figure 4.6: Examples of images generated by the GAN Module trained on a part of the LSUN-bedroom dataset and conditioned on “machine-generated” captions.

4.5 Conclusion

We explored the problem of conditional image generation using Generative Adversarial Networks with machine-generated captions. For this task, we built a pipeline to first generate captions for uncaptioned datasets and then to use the “machine-generated” captions to condition a GAN. To test our pipeline, we run experiments on the LSUN-bedroom dataset, which is a subset of the LSUN dataset containing uncaptioned images of bedrooms, and then compare the generated images in the unconditional setting and in the conditional setting where “machine-generated” captions are used. The results observed in the experiments do not achieve success in the task of conditioning with “machine-generated” captions. So we identify, analyze, and propose possible solutions to the obstacles that need to be overcome.

The Image Captioning Module that we trained on the COCO dataset seems to generate captions too similar to each other. Moreover, the captions we generated lack details

	Generated captions: • a bedroom with a bed in a room with a tv • a bedroom with a bed in a room with a table • a bedroom with a bed in a room with a window • a bedroom with a bed in a room with a television • a bedroom with a bed in a room with a room
	Generated captions: • a man standing in a bed in a room • a man standing in a bed in a room with a bed • a woman standing in a bed in a room • a man standing in a bed in a room with a television • a bedroom with a bed in a room with a table

Table 4.1: Examples of machine-generated captions for two images taken from LSUN-bedroom.

and contain some errors. This is probably related to the fact that more diverse and detailed captions are needed during training in order to achieve significant improvements. During a subsequent review of works on captioning, we found a work from Shetty et al. [253], that promises to generate more different captions, instead of variations of the same caption. This result is achieved by using GANs for image captioning instead of other traditional methods. An open question is whether with a bigger dataset the GAN could learn the image-captions correspondence, even when captions are very similar for each image. We believe improving the quality of the generated caption is the main challenge for our method. An hybrid approach could make our proposed method work by making humans write captions on a subset of the dataset, then use the obtained captions to train a captioning system. For generating human captions, crowdsourcing platforms like Amazon Mechanical Turk (AMT) could be used. We are currently working on this idea because it is likely that it will lead to improvements in the quality of generated bedroom images, given that AMT could make it possible to have high-quality and more diverse captions. Moreover, we are also considering the use of the Fréchet Inception distance [101] to evaluate the generated captions and images.

II

Can we reduce the generalization gap by customizing the training objective?

5

Video question answering supported by a multi-task learning objective

5.1 Introduction

Video Question Answering (VideoQA) is a task that requires to analyze and jointly reason on both the given video data and a visual content-related question, to produce a meaningful and coherent answer to it ([120]). By solving this task, a computational model could reach human-level capabilities when dealing with both complex video and textual data, since it would require learning to reason about the elements of interest in the video and their spatial and temporal interactions related to the given question. VideoQA represents thus a challenging task at the interface between Computer Vision and Natural Language Processing (NLP) ([308, 79, 73]).

Typically, a VideoQA architecture consists of a video encoder, a text encoder, a fusion module, and a decoder to produce the final answer ([74]), as can be seen in Fig. 5.1.a. These components for VideoQA are often built from neural networks which are the outcome of research work both from the NLP and Computer Vision communities. Deep convolutional networks originally proposed for Computer Vision tasks, such as image classification or action recognition, are usually employed as the backbone of the video encoder: as an example, among the many proposed architectures, appearance features are computed by means of VGG ([259]) in [72, 73], while [74, 120, 111, 124] adopt ResNet ([98]); on the other hand, motion features can be produced by using C3D ([276]) (e.g. in [72, 73, 120, 124]) or BN-Inception ([116]), as in [74]. Similarly, text encoding involves the usage of word embedding techniques, which are algorithms that transform natural language words into fixed-size representations. Considering that these representations are suitable for neural network training, these techniques are responsible for the great developments in the NLP community in recent years, e.g. [251, 182] for the task of named entity recognition, and [53, 176] for text question-answering. Although

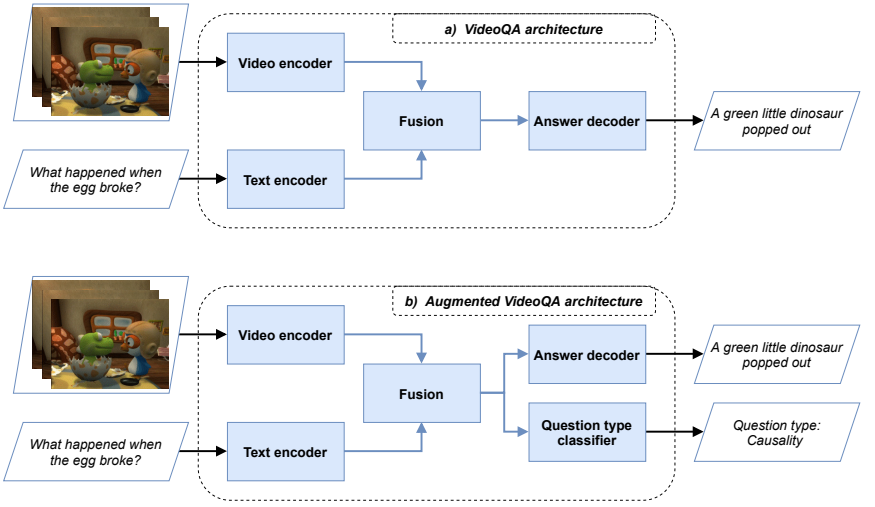


Figure 5.1: High level representation of a typical VideoQA architecture (shown in the upper part), consisting of: Video encoder and Text encoder which transform the raw input data into fixed-size representations; a Fusion module which combines the multi-modal information; and the Answer decoder, which computes the final answer. In the lower part, we present an augmented VideoQA architecture which leverages a multi-task learning strategy in order to jointly classify and answer the input question.

several word embedding techniques with different characteristics have been proposed in the literature, VideoQA architectures rely on only a few of these techniques, such as GloVe proposed by [214] and word2vec by [196] ([271]). As a consequence this language component, which provides the basis for the training process, is often underexplored in VideoQA architectures. Moreover, to the best of our knowledge, there are no complete and in-depth studies about the interaction between word embedding techniques and the VideoQA task. Hence, in this Chapter we propose an in-depth and extensive analysis to address these shortcomings.

Moreover, a multi-task learning strategy for VideoQA is introduced. As explained in a recent survey by [337], multi-task learning is a learning paradigm which aims at jointly learning multiple related tasks – in this way, the model needs to extract representations which are useful for all the considered tasks, therefore possibly leading to better generalization. This approach led to considerable improvements when applied to NLP (e.g. [216, 290]) and computer vision (e.g. [267, 326]), but also at the intersection of the two domains, especially when dealing with large scale visual-textual pretraining (e.g. [180]; [309]). Few works in the literature introduce auxiliary tasks designed for VideoQA, such as the one by [136], where the model is trained to perform question answering, as well as video-subtitle alignment and temporal localization. In this Chapter, an auxiliary task is introduced and it is designed with reference to the insights gathered from the aforementioned analysis of the word embedding techniques in the VideoQA domain.

In this Chapter, we propose a twofold contribution to VideoQA: firstly, a detailed analysis of word embedding techniques and of the final performance achieved by the

model; secondly, a novel multi-task learning strategy to train a VideoQA architecture which aims at improving its generalization capabilities. In particular, we consider four word embedding techniques: GloVe, a popular technique which leverages co-occurrence statistics to compute low-dimensional embeddings; ELMo ([215]), a technique which uses character-level convolutions and LSTM networks; BERT ([53]) and XLM ([148]) which leverage Transformers ([279]) as part of their encoding process. We integrate and evaluate these four techniques into three different VideoQA architectures, each of which adopts multiple state-of-the-art techniques. As the main and most relevant result of our analysis, we observe that different word embedding techniques perform differently when facing specific question types. With the term 'question type' we refer to a categorization of the questions based on the target of the question itself. As an example, the question type 'Causality' refers to questions that ask to identify an event which happens in relation to another one. As can be seen in Figure 5.1.a, a question of this type could involve a specific event such as "what happened when the egg broke?", to which the model may correctly answer by pointing out what happened next, e.g. "a green little dinosaur popped out". As detailed in the experimental analysis, we observe that BERT and XLM exhibit a higher accuracy (with a sensible margin) than ELMo and GloVe when dealing with 'Causality' questions. To investigate the relation occurring between word embedding techniques and question types, we propose a solution involving multi-task learning (Figure 5.1.b) which, differently from traditional approaches to VideoQA, jointly optimizes both the task-oriented loss and a novel classification loss related to the question types.

The main contributions of this Chapter can be summarized as follows:

- we integrate four of the most adopted word embedding techniques (GloVe, ELMo, BERT, and XLM) in three recent VideoQA architectures, from an attention-based encoder-decoder baseline ([120]) to more complex architectures involving memory ([79]) and reasoning ([73]);

- by quantitatively analyzing all the 12 combinations of embedding techniques and VideoQA architectures, we observe that word embedding techniques work better for specific question types;

- we propose a simple yet effective multi-task learning strategy which can help the considered models achieve better generalization, leading to considerable improvements on two public datasets;

- we release code and pretrained models to support research in this important field, to ease the reproducibility of the results, and to provide a codebase adaptable to different VideoQA datasets and models.

The rest of the Chapter is organized as follows. In Section 5.2 a comprehensive literature review is performed in order to contextualize the proposed method. The proposed methodology is presented in detail in Section 5.3. Several experimental results are shown and discussed in Section 5.4, concerning both the analysis of multiple word embedding techniques and the proposed multi-task learning strategy. Finally, Section 5.5 concludes the Chapter.

5.2 Related work

In this section we discuss the work related to the two main topics involved in this Chapter, i.e. Video Question Answering, and word embedding techniques.

5.2.1 Video Question Answering

Thanks to the recent availability of several large scale VideoQA datasets, such as TVQA ([153]), How2VQA69M ([315]), TGIF-QA ([120]), MSRVTT-QA and MSVD-QA ([308]), this task has gained more and more attention by researchers in both Computer Vision and NLP fields [308, 79, 73, 120, 72, 317, 315, 111, 210]. In particular, two types of tasks are often linked to VideoQA: the "open-ended" (e.g. in [120, 72, 308, 315]) and the "multiple choice" task (e.g. [120, 72, 139, 273, 153]). The former is usually treated as a classification problem where the correct answer is identified in a predefined set of possible answers, although it can also be approached through generative techniques by generating a free-form response word-by-word, e.g. in [339, 306]; the latter (i.e. "multiple choice", which is also the task that we tackle within this Chapter) involves the usage of a small pool of candidate answers (e.g. five choices in [120, 72]) which are possibly different for every question, and the model selects one of the candidates based on a score computed through a regressor. Considering that in this Chapter we describe and apply our approach in relation to the multiple choice task, in the following we focus on this specific task.

A prominent research direction for the multiple choice task consists in the usage of deep neural networks to learn suitable temporal or spatio-temporal features, eventually adopting attention mechanisms to filter out irrelevant features or redundant frames or frame regions, e.g. [120, 79, 153, 163]. "ST-VQA" (by [120]) integrated a temporal attention module to attend to the most important frames in the input clip, while leveraging LSTM networks to model the sequential aspect of both the visual and textual data. Conversely, [163] proposed a Positional Self-Attention block (based on [279]) to replace recurrent networks, while also using self-attention to learn self-attended single-modality features and a cross-modal attention mechanism in order to compute rich representations for the available visual and textual data. Although these methods led to considerable improvements on several public benchmarks, an important drawback is that they relied on clip- or frame-level representations, therefore missing out finer-grained details at the object-level. A recent research direction which focused on this aspect explored local relations between the visible objects and their natural language description . [111] built a complete graph using frame- and object-level features as node descriptors, making the graph location-aware by augmenting the nodes by means of spatial and temporal position features , and then reasoned over this structure with a Graph Convolutional Network ([142]). Yet, only visual information are used to build and reason on the graph structure. [124] argued that visual and linguistic factors have coordinated semantics which can be aligned to perform cross-modal reasoning, hence leading to the construction of an heterogeneous data structure. Although multiple video modalities are used by Jiang and Han, the semantic relations between them are not fully used. Therefore, [210] suggested to model both the visual-linguistic interactions as well as the semantic relations between different video modalities (e.g. appearance, motion) by using the question as a proxy. Differently from these works, a new research direction shifted the

attention to the training strategy. In particular, objective functions taken from the NLP domain were adapted to the VideoQA task. [317] performed masked language modeling (MLM) and next sentence prediction using object-level and question features as one of the inputs, while the candidate answers are used as possible next sentences. Similarly, [315] suggested using MLM and a contrastive objective in order to choose the correct answer using similarity metrics. Several papers (e.g. [151, 349, 157]) have also achieved notable performance on the target dataset by performing a large scale pretraining phase on large scale multi-modal datasets such as VisualGenome ([145]), HowTo100M ([195]), or How2VQA69M ([315]) by using language-only, vision-only, or language-vision proxy tasks. Yet, a major drawback of these pretraining procedures is the prohibitive computational cost, e.g. the training procedure on How2VQA69M lasted 2 days while using 8 Tesla V100 GPUs, according to [315]. Finally, given the sequential nature of the data involved in VideoQA, the usage of memory layers has also been explored, raising the possibility to interact with a memory made of multiple vectors, which is typically not possible in other neural networks which have a memory consisting of a single vector. "CoMem" ([79]) used memory layers to generate attention cues starting from both motion and appearance features. "HME-VQA" ([73]) introduced an heterogeneous memory layer while also proposing a multi-step LSTM-based reasoning technique. In this Chapter, among all the aforementioned solutions, we chose to use ST-VQA, CoMem, and HME-VQA because they offer increasingly complex and rich solutions which cover multiple state-of-the-art techniques, while also offering open source code bases. In particular, ST-VQA offers an attention-based encoder-decoder, CoMem also employs memory layers to support the generation of attention cues from both video modalities, and finally HME-VQA integrates multi-step reasoning as well. Although these works use advanced techniques to perform the video modeling or to fuse heterogeneous types of information, they only explore one technique to embed the words into vectorial representations, that is GloVe, therefore ignoring recent advancements in NLP. To this end, given that understanding the question is fundamental to predict the correct answer, in this Chapter we analyze how four popular embedding techniques interact with the network architectures used for VideoQA. Then, we propose a multi-task learning strategy to improve the generalization capabilities of a VideoQA system, by designing an auxiliary task based on the results of the preliminary analysis.

5.2.2 Word embedding techniques

NLP has rapidly evolved during the past few years and one of the most investigated topics is related to neural language models (LM). Before the introduction of BERT, GloVe and ELMo were two of the most used techniques.

[214] introduced GloVe, which is a *static*, non-contextual word embedding technique which computes the word vectors by leveraging both local and global statistics (e.g. word co-occurrence), assigning the same embedding (i.e. a real-valued vector) to a word independently from the context in which such word is used. Differently from GloVe, [215] proposed a contextual (i.e. each word receives an embedding depending on the context) LSTM-based LM called ELMo. Moreover, the objective of ELMo is to estimate the probability distribution of the training corpus using recurrence, and is thus classified as an autoregressive LM.

Lately, the introduction of BERT by [53] and its variants (e.g., XLM by [148] and

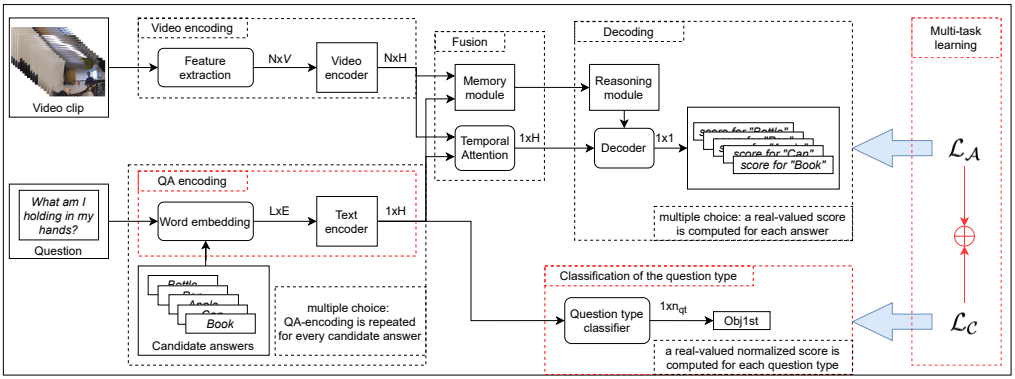


Figure 5.2: General architecture of the models considered in this Chapter, which focuses on the Word Embedding module and the Question type classifier (outlined in red). The former receives the question and the candidate answers, and outputs L embeddings of size E. The latter is trained in a multi-task learning style and we show it helps improving the performance.

DistilBERT by [242]) showed that these models have strong transfer learning capabilities by simply attaching and training a task-specific head over the pretrained backbone. Being based on Transformers ([279]), they are solely based on attention mechanisms and do not use any recurrent neural network. Because of this they are classified as autoencoding LMs: instead of estimating the probability distribution of the corpus, they learn a function to reconstruct the input from masked versions of it.

With the introduction of BERT, several tasks in NLP reached new state-of-the-art results, yet its predecessors are still used in many works, e.g. GloVe in [73, 111, 210, 282, 124]. As mentioned before, to fairly analyze the influence of the most important word embedding techniques in the VideoQA task, we propose a study to understand which one to use by integrating each of them in three VideoQA models.

5.3 Methodology

In this Chapter we explore the usage and integration of several word embedding techniques into three different VideoQA architectures (i.e. ST-VQA, CoMem, and HME-VQA) which involve multiple state-of-the-art techniques including attention mechanisms, memory layers, and multi-step reasoning. To treat them all in a shared but comprehensive manner, we present in Figure 5.2 a detailed overview of a VideoQA architecture: it comprises both the common components, such as "Feature extraction" and "Word embedding", as well as the modules which are exclusive to only some of the architectures, such as the "Reasoning module" which is only used by HME-VQA. Moreover, on the right we also outline the additional components related to the proposed multi-task learning strategy, including a module used to classify the question type, and the joint loss function (built on $\mathcal{L}_\mathcal{A}$ and $\mathcal{L}_\mathcal{C}$, described in Section 5.3.1). As already mentioned, a VideoQA architecture can be seen as made of four blocks, that

is Video encoding, QA encoding, Fusion, and Decoding. Given the input data, we extract a sequence of embeddings for both the video (in "Feature extraction") and the input question (in "Word embedding"), as well as for the candidate answers. For the video, VGG is employed to extract appearance features, while C3D is used for motion features. For the textual data, we use one of the word embedding techniques that we explore in this Chapter (see Sec. 5.4.2 for more details). These two steps are done for all the architectures that we consider, which are ST-VQA ([120]), CoMem ([79]), and HME-VQA ([73]). Then, for both visual and textual data, we employ an encoder made of two stacked LSTM networks to model the evolution of the features. Note that ST-VQA concatenates appearance and motion features before processing them by using the Video encoder; CoMem and HME-VQA independently model the two sequences of features via two independent Video encoders which follow the same structure. The Fusion block aims at computing a representation which takes both the video and textual information into account. In ST-VQA, this is done through a Temporal Attention module, which weighs each visual features vector based on the aggregated textual representation; in CoMem, appearance and motion features are used to provide attention cues to each other by employing a Memory module; finally, a Reasoning module is used in HME-VQA to compute an aggregated representation of the output of the heterogeneous memory layer, while also employing two temporal attention modules to compute modality-independent attention-weighted vectors (see Sec. 5.3.2 for an in-depth explanation). The fused features are then used in the decisional process to predict a regression score for the candidate answer. Note that in the case of HME-VQA the input to the decoder uses both the attended visual vectors and the output of the Reasoning module. To optimize the network parameters, an hinge loss is used to enforce a margin (e.g. 1, as in Eq. 5.2) between the score computed for the correct answer and all the other candidate answers.

In Section 5.3.1 we thoroughly describe the proposed multi-task learning strategy. For completeness, we also provide further details about the adopted methods in Section 5.3.2, by focusing on their differences.

5.3.1 Multi-task learning strategy

When asking a question to a VideoQA model, we expect it to extract visual and textual information which are related to the question itself. Furthermore, we expect questions of the same category to share a similar visual and textual joint representation as computed by the Fusion module. As an example, questions asking to identify an object may require spatial features which are closely related to the objects shown in the video, while asking to recognize an action may shift the focus on temporal aspects. For this reason, we propose to incorporate the question type (as a classification objective) into the loss function we strive to optimize.

The proposed multi-task learning strategy involves a joint loss function, comprising of a pairwise hinge loss $\mathcal{L}_{\mathcal{A}}$, which is used to train the model for the VideoQA multiple choice task, as it is often done in the literature (e.g. in [120, 72]) and a classification loss $\mathcal{L}_{\mathcal{C}}$ which we use to make the model able to categorize an input question into one of the predetermined types. Such a joint loss can be described as:

$$\mathcal{L} = \mathcal{L}_{\mathcal{A}} + \mathcal{L}_{\mathcal{C}} \tag{5.1}$$

For a given input sample, the pairwise hinge loss can be described as:

$$\mathcal{L}_{c,r} = \begin{cases} 0 & \text{if } c = r \\ max(0, 1 + s_c - s_r) & \text{if } c \neq r \end{cases} \tag{5.2}$$

where s_c and s_r are the scores d_r computed by the Decoder (see Sec. 5.3.2, Eq. 5.11 for more details) for the candidate answer c and the right answer r. We use this loss function because it allows us to optimize the parameters of the models under analysis by enforcing a margin between the scores s_c and s_r. Moreover, by using it we were able to reproduce previously reported results ([72]). To compute $\mathcal{L}_{c,r}$ for each sample in the minibatch, we use the following equation:

$$\mathcal{L}_{\mathcal{A}} = \sum_{q \in \mathcal{Q}} \sum_{c \in \mathcal{C}_q} \mathcal{L}_{c,r} \tag{5.3}$$

where $\mathcal{Q}$ represents the questions in the minibatch, while $\mathcal{C}_q$ and r are respectively the set of candidate answers and the correct answer for q.

To deal with the additional classification objective $\mathcal{L}_{\mathcal{C}}$, we augment all the considered architectures by attaching a classifier head on top of the Text encoder:

$$l_{qt} = softmax(\epsilon_w W_{qt} + b_{qt}) \tag{5.4}$$

where $\epsilon_w \in \mathbb{R}^{1 \times H}$ is the output of the Text encoder (see Sec. 5.3.2 for more details), $W_{qt} \in \mathbb{R}^{H \times n_{qt}}$ and $b_{qt} \in \mathbb{R}^{1 \times n_{qt}}$ are trainable parameters, H is twice the hidden size h, and finally n_{qt} is the amount of question types in the considered dataset. To train the model for this additional task, we consider the following equations:

$$\chi(x, y) = \frac{1}{n_{qt}} \sum_{i=1}^{n_{qt}} -(y_i \cdot \log(x_i) + (1 - y_i) \cdot \log(1 - x_i)) \tag{5.5}$$

$$\mathcal{L}_{\mathcal{C}} = \frac{1}{|\mathcal{Q}| \cdot |\mathcal{C}_q|} \sum_{q \in \mathcal{Q}} \sum_{c \in \mathcal{C}_q} \chi(l_{qt}, \text{one-hot}(t)) \tag{5.6}$$

where t is the type of the question q, and one-hot(t) computes its one-hot representation. By using l_{qt} we consider the question as well as the candidate answer because both may contain helpful and discriminative information while optimizing for this task.

As previously mentioned, we apply our multi-task learning strategy to several different architectures, in order to show its general applicability. In the following section, we provide a more detailed presentation of the considered VideoQA architectures.

5.3.2 VideoQA architectures

Here we describe three VideoQA models which can be seen as made of four blocks ([74]): Question-Answer (QA) Encoding, Video Encoding, Fusion, and Decoding. This can be observed both in Fig. 5.1, where we depict it from a high level view, and in Fig. 5.2, which shows a general framework to cover all the models used in this Chapter. These three models involve several state-of-the-art techniques, including attention mechanisms,

memory layers, and multi-step reasoning, offering an heterogeneous experimental setting. In Figure 5.3 we also include a more in-depth view on the three architectures in order to highlight the major differences between them, which are also commented in the following subsections. The four blocks previously identified in Fig. 5.2 are respectively colored in purple, blue, yellow, and darker yellow. The proposed multi-task learning strategy (see Sec. 5.3.1) is highlighted in red. As can be seen, the auxiliary task introduced in this Chapter, that is the prediction of the question type, is performed by using the textual features computed by the LSTM-based Text Encoder. The proposed multi-task learning strategy is easily extendable to heterogeneous architectures and, in fact, in Sec. 5.3.1 it is shown how to apply it to three different techniques from the literature.

ST-VQA. The first model we use is based on ST-VQA proposed by [120], an encoder-decoder architecture supported by attention mechanisms. Since we deal with the multiple choice task, the QA encoding module receives a question and a pool of candidate answers. Let $q_1 \dots q_m$ and $a_1 \dots a_n$ be the sequence of m tokens of the question and n tokens of (one of the candidate) answer. As shown in Fig. 5.2, the encoding of question and candidate answer is performed for each of the candidates, since they are (possibly) different for each question. In Fig. 5.3 (left) this is shown as the "Textual question + candidate answer" block. To do so, $q_1 \dots q_m$ and $a_1 \dots a_n$ are concatenated into δ and used as input to the embedding technique (shown as "Word embedding" in Fig. 5.3), eventually adding some special tokens (for more details, see Sec. 5.4.2). Hence, the textual data are first embedded into $\phi_w \in \mathbb{R}^{L \times E}$, where L is the number of tokens in question and answer, and E is the embedding size. Then, ϕ_w is input to the Text Encoder, which consists of two stacked LSTM networks. The encoded textual features ϵ_w are obtained by concatenating the *last* hidden state of both the LSTM networks, thus forming a feature vector $\epsilon_w \in \mathbb{R}^{1 \times H}$. In the Video Encoding block, both motion and appearance features are obtained from an input video clip made of N frames. Both the feature extraction and the Video Encoder are depicted in Fig. 5.3 with the blue color. To compute the appearance features, they use a frozen VGG-16, pretrained on ImageNet, and extract the *fc7* activations ($\phi_a \in \mathbb{R}^{N \times 4096}$). To compute the motion features, they use a frozen C3D, pretrained on Sports1M ([132]) and fine-tuned on UCF101 ([265]), and extract the *fc7* activations ($\phi_m \in \mathbb{R}^{N \times 4096}$). In this Chapter, we use VGG-16 and C3D because the feature vectors are computed through a transformation of the feature maps computed by the convolutional layers, and not by employing a global pooling layer. In fact, while the usage of the latter operation (for example, employed in ResNet by [98]) greatly reduces the quantity of parameters in the model, it also leads to a loss of the positional information available in the activation tensors. The features extracted from VGG and C3D are concatenated obtaining $\phi_{a,m} \in \mathbb{R}^{N \times V}$ (with $V = 8192$) and then input to a Video Encoder made of two stacked LSTM networks. Despite similar in structure to the Text Encoder, the output of the Video Encoder consists of the full sequence of hidden states, i.e. $\epsilon_v \in \mathbb{R}^{N \times H}$.

ST-VQA features an attention-based ([11, 106]) Fusion block, shown in yellow in Fig. 5.3 (left), which lets the model learn which frames are more important based on the encoded textual features. It receives in input the encoded video features ϵ_v and the

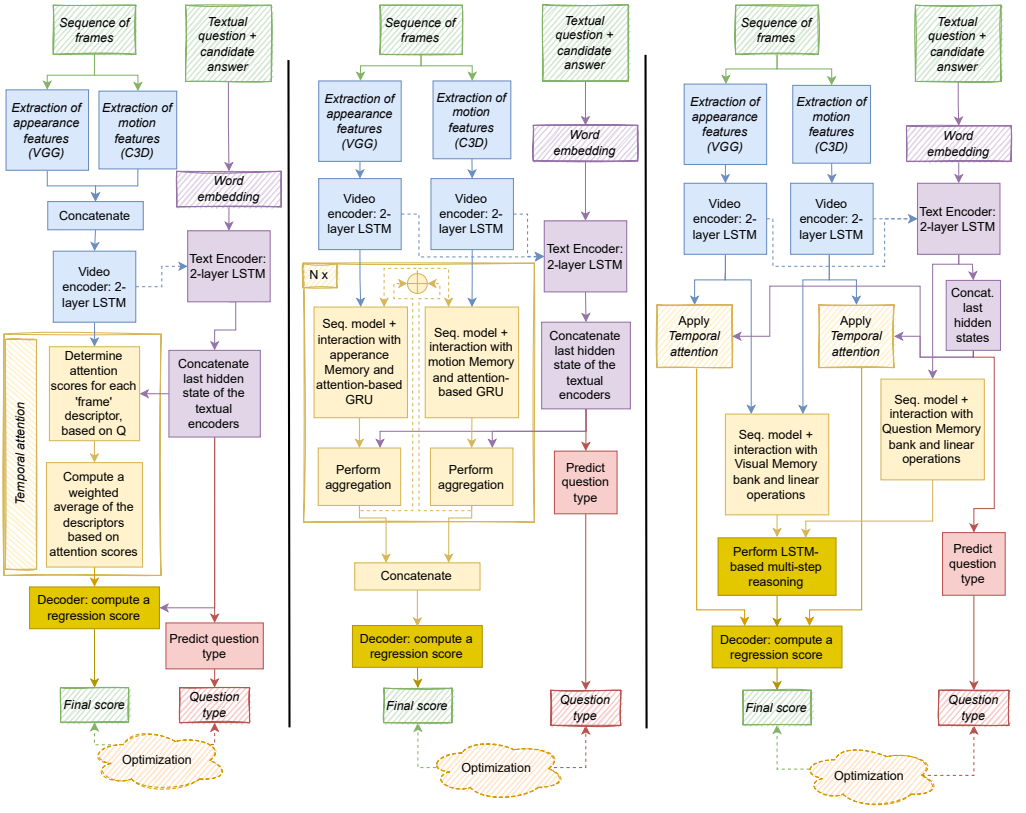


Figure 5.3: Detailed diagram of the three models we selected from the literature, *i.e.* (**left**) ST-VQA by [120], (**middle**) CoMem by [79], and (**right**) HME-VQA by [73]. Compared to Fig. 5.2, we color in blue the "Video encoding", in purple the "QA encoding", in yellow the "Fusion", in darker yellow the "Decoding", and finally we highlight in red the modification applied to the base algorithms in order to use the proposed multi-task learning strategy. In the "Optimization" cloud we perform $\mathcal{L}_{\mathcal{A}}+\mathcal{L}_{\mathcal{C}}$.

textual features ϵ_w, and can be described by the following equations:

$$\omega_s = tanh(\epsilon_v W_v + \epsilon_w W_w + b_s) W_s \tag{5.7}$$
$$\alpha_s = softmax(\omega_s) \tag{5.8}$$
$$\omega_a = \mathbb{1} \cdot (\alpha_s \circ \epsilon_v) \tag{5.9}$$

where $W_v, W_w \in \mathbb{R}^{H \times h}$, $W_s \in \mathbb{R}^h$, and $b_s \in \mathbb{R}^{1 \times h}$ are learnable parameters. Eq. 5.7 and 5.8 are used to compute the attention weights $\alpha_s \in \mathbb{R}^{N \times 1}$ for the N visual feature vectors and their interaction with the textual information; then, Eq. 5.9 implements a sum-pool operation, where $\mathbb{1}$ is a row of ones ($\mathbb{1}^{1 \times N}$), which is used to sum the weighted visual information. $\circ$ represents the element-wise multiplication operator.

Finally, the decoder we use is based on the one proposed in [72]. Decoding is done for each QA pair, that is in our multiple choice setting, five times with different textual features producing five different scores, one per candidate answer. It can be described by the following equations:

$$d_f = tanh(\omega_a W_a + b_a) \tag{5.10}$$
$$d_r = (d_f \circ \epsilon_w) W_d + b_d \tag{5.11}$$

where $W_a \in \mathbb{R}^{H \times H}$, $W_d \in \mathbb{R}^{H \times 1}$, $b_a \in \mathbb{R}^{1 \times H}$, and $b_d \in \mathbb{R}$ are parameters, $d_f \in \mathbb{R}^{1 \times H}$, $d_r \in \mathbb{R}$. d_r is the score obtained by testing a specific candidate answer (out of the five possible choices related to the given question). The Decoding step is shown with a darker shade of yellow in Fig. 5.3.

CoMem. The CoMem model is based on the work by [79]. As in ST-VQA the textual features are computed by the word embedding technique and the Text Encoder. The visual features are again extracted using VGG and C3D but, in this case, they are not concatenated and they are encoded with two independent Video Encoders. Furthermore, hidden and cell state of the Text Encoder are initialized with those of the Video Encoder. Yet, the main difference with the ST-VQA approach is the usage of a Memory module within the Fusion block, shown in yellow in Fig. 5.3 (middle), which is supported by a co-attention mechanism. That is, they show appearance features are useful to guide the extraction of relevant motion features, and vice versa. To capture these interactions, both attention and memories are exploited. Moreover, the Memory module is used sequentially in the architecture as a fusion technique by replacing the Temporal Attention. These operations are shown in Fig. 5.3 (middle) in yellow.

In particular, CoMem uses and iteratively updates two memories called "appearance memory" and "motion memory": at every iteration, both are updated by an attention function which jointly attends to both motion and appearance encoded features, the memory, and the question embedding. Then, appearance and motion features are used to update each memory (shown with the $\bigoplus$ operator in Fig. 5.3). This operation is repeated a fixed amount of times and is depicted with the "N x" block.

HME-VQA. As in CoMem, HME-VQA ([73]) follows a similar overall flow: the visual features computed by using VGG and C3D with independent Video Encoders, and the textual features are computed with the Text Encoder applied on top of the word embeddings. Differently from CoMem, HME-VQA uses two memories, a "visual memory" and a "question memory", as depicted in Fig. 5.3 (right). The former is

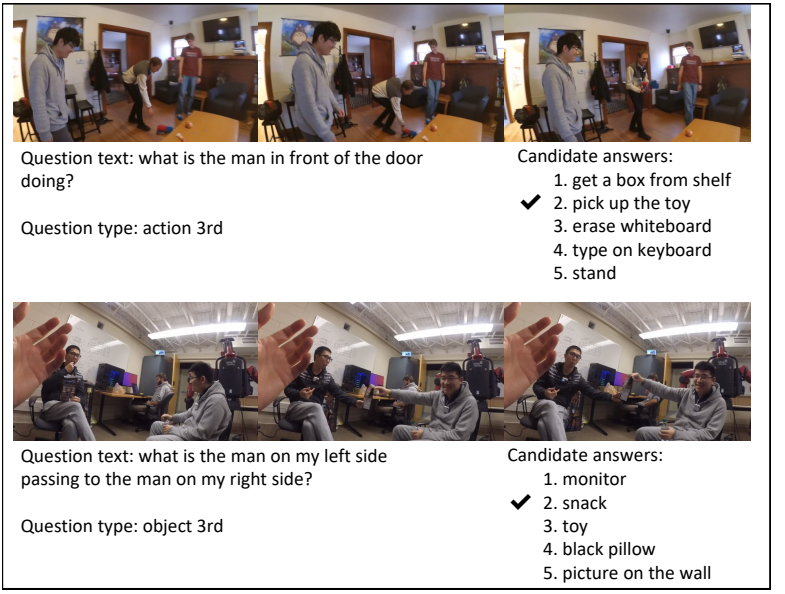


Figure 5.4: Samples of clips, questions, and candidate answers from EgoVQA.

updated by an attention function that exploits three hidden states, which consider appearance and motion features both separately and jointly. In the latter only one hidden state is used. A second novelty in HME-VQA is the usage of an LSTM-based Reasoning module (colored with a dark yellow in Fig. 5.3), which consists of three steps: first of all two context vectors, c_v and c_q, are created by attending to the hidden states of the "visual memory" and the "question memory", and the previous hidden state s_{t-1}; then c_v, c_q, and s_{t-1} are used to separately compute attention weights, which are used to compute the "fused knowledge", i.e. a weighted sum of c_v and c_q; finally, the LSTM updates s_t using the fused knowledge and s_{t-1}. The last hidden state of the LSTM is used as a distilled version of the given data . A Temporal attention module is separately applied on the appearance and motion features, as shown in Fig. 5.3. Finally, the text-attended visual features and the output of the Reasoning module are used in conjunction with the Decoder to compute the score for the answer.

5.4 Results and Discussions

To perform the analysis of the word embedding techniques and to validate our multi-task learning strategy, we choose to use two public VideoQA datasets, PororoQA ([139]) and EgoVQA ([72]), as they also briefly discuss question types. After presenting these datasets, we thoroughly describe the word embedding techniques that we used, and we discuss both the overall results and the per question type results.

Code	Description	Example
Act_{1st}	action performed by camera wearer	"what am I doing"
Act_{3rd}	action performed by different actor	"what is the man in red clothes doing"
Obj_{1st}	object the camera wearer is interacting with	"what am I holding in my hands"
Obj_{3rd}	object a different actor is interacting with	"what is placed on the desk"
Who_{1st}	who the camera wearer is interacting with	"who am I talking with"
Who_{3rd}	who is performing a certain action	"who is eating salad"
Cnt	number of persons or objects in the scenes	"how many people am I talking with"
Col	identify the color of an object	"what is the color of the toy in my hands"

Table 5.1: Description of the question types available in the EgoVQA dataset.

Code	Description	Example
Abs	questions about abstract concepts	"what is the weather like in the forest"
Act	recognize the action performed	"what are Pororo and Crong doing"
Caus	describe which event follows another one	"what happened when the egg broke"
Det	detail of something in the clip	"what kind of bird is Pororo"
Loc	describe where an event takes place	"where did Pororo take the egg"
Met	describe *how* something is done	"how did Pororo introduce himself"
Per	identify who did a specific action	"who was sliding on the ice"
Reas	motivate a specific event	"why is Pororo running away"
Stmt	questions about the content of a speech	"what did the baby dinosaur say first"
Time	describe when an event takes place in the clip	"when does Pororo find an egg"
Y/N	yes/no questions with an explanation	"are Pororo's friends scared of the dinosaur"

Table 5.2: Description of the question types available in the PororoQA dataset.

5.4.1 The datasets

EgoVQA Presented in [72], it features more than 600 QA pairs and the same number of clips, which are 20-100 seconds long and are obtained from 16 egocentric videos (5-10 minutes long) based on 8 different scenarios. An example of these egocentric videos and QA pairs can be seen in Fig. 5.4. The questions can be grouped in eight types, as described in Table 5.1.

For each video and question five candidate answers are provided, of which only one is correct. The wrong answers are randomly sampled from a candidate pool based on the question type, i.e. if the question requires to recognize an action, the five candidates (the right one and the four wrong) are actions.

PororoQA Introduced by [139], it features around 8,800 QA pairs over 6,160 clips, which are 3.5 seconds long (on average) and are obtained from 166 episodes of the Korean cartoon "Pororo". PororoQA follows the multiple choice setting with five candidates, and the question types are shown in Table 5.2. Although this dataset also offers scene descriptions and subtitles, we choose not to use them because it would be a different task (Video Story Question Answering, see [273, 153, 80] as well), and thus out of the scope of this Chapter. For this reason, we are only using RGB frames, hence why we are not comparing the performance results we obtain with [139], [138], and [317], where scene descriptions and subtitles are exploited as well.

5.4.2 Word embeddings

In our experiments, we choose to use GloVe, ELMo, BERT, and XLM because of their popularity and because they provide both contextual and non-contextual embeddings. Note that they use different tokenizers: in particular we use full words for GloVe and ELMo, WordPiece ([303]) for BERT, and Byte-Pair Encoding ([249]) for XLM.

GloVe. By using GloVe, pretrained on the Common Crawl dataset, a vector of size $E = 300$ is computed for each word in both question and answer. Since GloVe is not contextual, question and answer can be given in input to it either jointly or separately obtaining the same embedding. In the former case, the input to GloVe is a simple concatenation of the tokens, i.e. $\delta = q_1 \dots q_m a_1 \dots a_n$, whereas the output is $\phi_w \in \mathbb{R}^{(n+m)\times E}$. In the latter case, two embedded representations are computed by separately using GloVe on $q_1 \dots q_m$ and $a_1 \dots a_n$, which are then concatenated to obtain ϕ_w.

ELMo. ELMo is a contextual word embedding technique based on LSTMs which computes for each word multiple representations, derived from its hidden states. In our setting, we extract the topmost representation of size $E = 1024$. Since ELMo is contextual, as opposed to GloVe which is not, the word embeddings for question and answer need to be jointly computed, i.e. the input to ELMo is $\delta = q_1 \dots q_m a_1 \dots a_n$, with $|\delta| = L$.

BERT. Similarly to ELMo, BERT computes multiple representations for each word. We use the base version consisting of 12 attention heads and 12 layers, each of which produces a different embedding of size $E = 768$. We use the embeddings from the last layer. For BERT, $\delta = \alpha q_1 \dots q_m \sigma a_1 \dots a_n \sigma$, where α is the token '[CLS]', and σ is the separator '[SEP]'.

Note that although BERT already provides an aggregated output in the representation of the '[CLS]' token, we chose to also adopt the LSTM-based Text Encoder (see Sec. 5.3.2) on top of it because of two reasons: firstly, to have an overall similar structure across all four embedding techniques; secondly, because it can provide further context while also improving the final performance ([80]).

XLM. XLM is a variant of BERT which uses a different training technique and also uses BERT as an initialization step for machine translation models. In particular, we adopt the base version of XLM, which uses 12 layers and 16 attention heads. The word embeddings computed using this method have size $E = 2048$. δ is defined in the same way as for BERT.

5.4.3 Evaluation protocol

In our setting, we fix $H = 512$ and $h = 256$. To optimize the parameters we use Adam ([141]) with a fixed learning rate of 10^{-3} and a batch size of 8.

To implement our solution we use Python 3.6, Numpy 1.18, and PyTorch 1.7. We use AllenNLP ([81]) to test ELMo[1], and the 'transformers' library 3.5.1 ([297]) to test BERT[2] and XLM[3].

[1] pretrained: 'elmo_2x4096_512_2048cnn_2xhighway_weights'
[2] pretrained: 'bert-base-uncased'
[3] pretrained: 'xlm-mlm-en-2048'

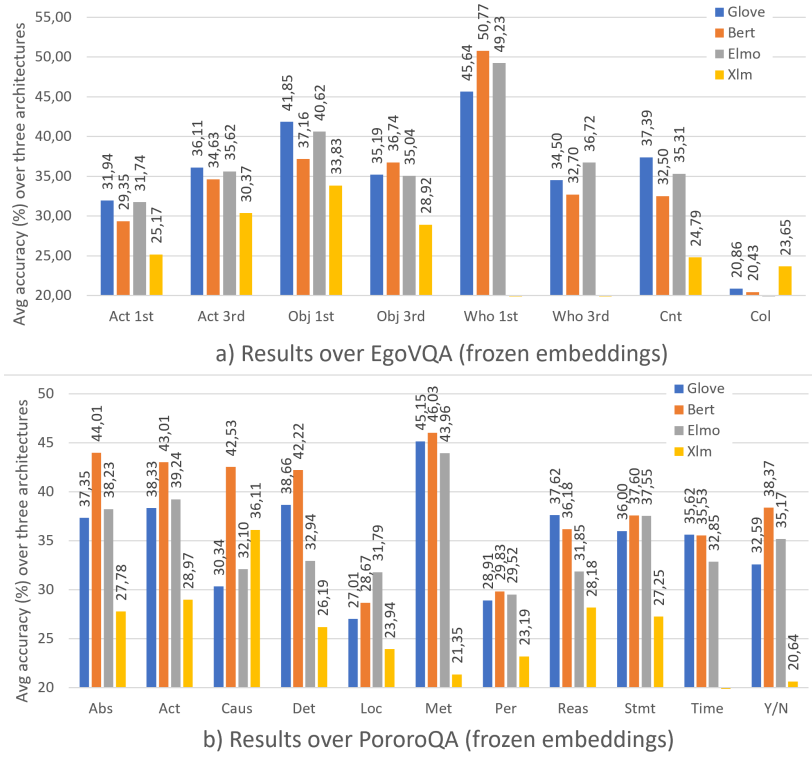


Figure 5.5: We report for each question type the average accuracy (setting the minimum to the random chance, i.e. 20%) obtained by using a specific word embedding technique. It is possible to see that different question types are best dealt with by using different embeddings. Best viewed in color.

To evaluate the performance of the multiple combinations explored in this Chapter, we train for 20 epochs, then we select the model with the best validation accuracy and use it for testing. This is done five times (fixed seeds, 0 to 4), in order to obtain more stable and reliable results. It is particularly important for EgoVQA, where the amount of available data is relatively low and thus susceptible to highly variable results over multiple runs.

5.4.4 Results using the frozen embeddings

The first set of results analyze how different embedding techniques affect the final performance, while using them pretrained and frozen. We start from this experiment because word embeddings are often kept frozen and not trained, e.g. in [296], since they are learned on big text corpora from which the embeddings gather semantics which can transfer well to downstream tasks. Tables 5.3 and 5.4 present the overall accuracy for EgoVQA and PororoQA, and show that BERT provides the best embeddings for the task: in particular, adopting BERT in the ST-VQA architecture leads to an average

	GloVe	ELMo	BERT	XLM
ST-VQA	$35.7_{\pm 4.2}$	$34.9_{\pm 4.5}$	$\mathbf{36.1_{\pm 6.0}}$	$24.0_{\pm 3.3}$
CoMem	$34.1_{\pm 3.8}$	$33.6_{\pm 4.8}$	$31.6_{\pm 5.3}$	$25.7_{\pm 3.9}$
HME-VQA	$35.4_{\pm 3.1}$	$35.5_{\pm 3.8}$	$33.6_{\pm 4.5}$	$26.8_{\pm 3.2}$

Table 5.3: Average accuracy over EgoVQA using the frozen embeddings.

	GloVe	ELMo	BERT	XLM
ST-VQA	$36.8_{\pm 0.5}$	$35.7_{\pm 0.9}$	$\mathbf{39.7_{\pm 0.9}}$	$27.4_{\pm 1.2}$
CoMem	$33.3_{\pm 1.1}$	$35.4_{\pm 1.0}$	$36.1_{\pm 0.8}$	$22.2_{\pm 0.7}$
HME-VQA	$36.8_{\pm 1.8}$	$34.5_{\pm 1.1}$	$39.2_{\pm 0.7}$	$27.8_{\pm 0.9}$

Table 5.4: Average accuracy over PororoQA using the frozen embeddings.

accuracy of 36.1% in EgoVQA (Table 5.3) and 39.7% in PororoQA (Table 5.4). But, especially for EgoVQA, Table 5.3 shows that other techniques can provide useful embeddings depending on the architecture chosen: as an example, ST-VQA with GloVe achieves 35.7%, while HME-VQA with ELMo obtains 35.5%.

To perform a finer-grained analysis, we present in Tables 5.5 and 5.6 the accuracy values based on the question type for EgoVQA and PororoQA. We perform this analysis because, as previously mentioned, question types may represent a key element when trying to answer a question. As an example, Table 5.5 shows that, when dealing with Act_{3rd} questions, HME-VQA with GloVe (row identified with "H;G") achieves 41.5% average accuracy, yet when using BERT loses around 6% (row "H;B" shows 35.2%). Similarly, CoMem with GloVe (row "C;G") has an accuracy of 40.3% when answering Cnt questions, yet it only obtains 32.8% when using BERT. Similar differences can be also observed on PororoQA in Table 5.6, e.g. when faced with $Reas$ questions, ST-VQA obtains 39.5% and 31.0% when adopting, respectively, GloVe (row "S;G") or ELMo (row "S;E"). Thus, by analyzing the results while also taking the question type into account can lead to some insights which we detail in the next subsections.

EgoVQA

In Figure 5.5 we propose two plots where we present the accuracy obtained when averaging with respect to the architecture used. For example, in Fig. 5.5.a the blue column (related to GloVe) over Obj_{1st} shows 41.85, which is the mean value of the average accuracy obtained by ST-VQA, CoMem, and HME-VQA for that type of question. We do this in order to have a simplified view of the detailed results shown in Tables 5.5 and 5.6.

In the case of EgoVQA, Fig. 5.5.a shows that on average GloVe and ELMo achieve better performance than BERT and XLM.

Moreover, we can also observe that ELMo obtains a 2.2% margin over GloVe when trying to identify who is performing a given action in front of the camera wearer (Who_{3rd}). Considering that the questions of this type, Who_{3rd}, are longer with respect to other types (11.5 words versus an average of 9.6), this may be due to ELMo having the memory capabilities provided by the LSTMs while also exploiting the bidirec-

Met;Emb	Question type accuracy (%)							
	Act_{1st}	Act_{3rd}	Obj_{1st}	Obj_{3rd}	Who_{1st}	Who_{3rd}	Cnt	Col
S; G	$29.2_{\pm 2.2}$	$33.9_{\pm 2.2}$	$42.6_{\pm 4.1}$	$37.0_{\pm 4.9}$	$\mathbf{52.3_{\pm 3.1}}$	$40.3_{\pm 5.4}$	$38.8_{\pm 2.5}$	$20.0_{\pm 4.3}$
S; E	$29.2_{\pm 1.8}$	$32.6_{\pm 3.4}$	$41.1_{\pm 4.6}$	$35.6_{\pm 2.2}$	$49.2_{\pm 3.8}$	$41.6_{\pm 2.1}$	$38.4_{\pm 3.6}$	$18.1_{\pm 4.4}$
S; B	$28.9_{\pm 2.8}$	$35.9_{\pm 1.8}$	$39.6_{\pm 3.2}$	$35.6_{\pm 2.8}$	$\mathbf{52.3_{\pm 3.1}}$	$\mathbf{45.4_{\pm 2.1}}$	$33.1_{\pm 5.0}$	$20.0_{\pm 5.9}$
S; X	$24.8_{\pm 4.1}$	$26.8_{\pm 3.4}$	$31.5_{\pm 5.6}$	$31.4_{\pm 5.9}$	$12.3_{\pm 6.2}$	$9.5_{\pm 2.8}$	$22.8_{\pm 5.8}$	$21.3_{\pm 4.8}$
C; G	$30.7_{\pm 3.2}$	$33.0_{\pm 4.4}$	$\mathbf{45.5_{\pm 4.0}}$	$35.1_{\pm 1.4}$	$40.0_{\pm 7.5}$	$27.6_{\pm 13.8}$	$\mathbf{40.3_{\pm 0.6}}$	$\mathbf{23.2_{\pm 6.6}}$
C; E	$28.4_{\pm 2.1}$	$34.4_{\pm 1.9}$	$43.3_{\pm 4.5}$	$35.1_{\pm 2.9}$	$50.8_{\pm 3.8}$	$31.7_{\pm 3.9}$	$35.6_{\pm 4.6}$	$17.4_{\pm 7.8}$
C; B	$26.3_{\pm 2.8}$	$32.8_{\pm 1.5}$	$38.5_{\pm 3.2}$	$35.1_{\pm 2.5}$	$50.8_{\pm 3.8}$	$22.9_{\pm 4.1}$	$32.8_{\pm 7.0}$	$21.9_{\pm 3.8}$
C; X	$21.8_{\pm 4.1}$	$32.2_{\pm 3.1}$	$35.2_{\pm 5.2}$	$29.8_{\pm 3.3}$	$23.1_{\pm 8.4}$	$16.2_{\pm 2.7}$	$19.4_{\pm 4.7}$	$\mathbf{23.2_{\pm 7.7}}$
H; G	$35.8_{\pm 3.1}$	$\mathbf{41.5_{\pm 4.0}}$	$37.4_{\pm 4.4}$	$33.5_{\pm 2.3}$	$44.6_{\pm 5.8}$	$35.5_{\pm 10.1}$	$33.1_{\pm 5.4}$	$19.3_{\pm 3.5}$
H; E	$\mathbf{37.6_{\pm 4.0}}$	$39.8_{\pm 1.3}$	$37.4_{\pm 5.9}$	$34.4_{\pm 5.2}$	$47.7_{\pm 5.8}$	$36.8_{\pm 7.2}$	$31.9_{\pm 6.5}$	$20.6_{\pm 8.3}$
H; B	$32.8_{\pm 3.0}$	$35.2_{\pm 3.3}$	$33.3_{\pm 3.7}$	$\mathbf{39.5_{\pm 3.0}}$	$49.2_{\pm 6.2}$	$29.8_{\pm 8.0}$	$31.6_{\pm 7.3}$	$19.4_{\pm 8.4}$
H; X	$25.7_{\pm 4.3}$	$30.7_{\pm 3.3}$	$37.0_{\pm 4.5}$	$29.5_{\pm 1.6}$	$23.1_{\pm 10.9}$	$14.3_{\pm 5.9}$	$26.6_{\pm 4.4}$	$19.3_{\pm 5.4}$

Table 5.5: Accuracy per question type over EgoVQA using the frozen embeddings. We introduce "Act_{1st}", etc. in Table 5.1 to identify the question types. S, C, H, G, E, B, X represent respectively ST-VQA, CoMem, HME-VQA, GloVe, ELMo, BERT, and XLM.

M;E	Question type accuracy (%)										
	Abs	Act	Caus	Det	Loc	Met	Per	Reas	Stmt	Time	Y/N
S;G	$38.5_{\pm 1.7}$	$39.8_{\pm 2.0}$	$33.2_{\pm 3.9}$	$37.5_{\pm 3.7}$	$28.7_{\pm 4.6}$	$45.4_{\pm 1.9}$	$30.3_{\pm 0.9}$	$\mathbf{39.5_{\pm 4.3}}$	$37.9_{\pm 2.8}$	$37.1_{\pm 4.1}$	$32.9_{\pm 3.3}$
S;E	$40.2_{\pm 2.2}$	$39.3_{\pm 1.6}$	$33.3_{\pm 5.2}$	$34.8_{\pm 4.1}$	$29.9_{\pm 5.7}$	$43.7_{\pm 2.5}$	$30.8_{\pm 1.1}$	$31.0_{\pm 5.5}$	$40.2_{\pm 1.9}$	$28.6_{\pm 5.4}$	$39.3_{\pm 3.0}$
S;B	$\mathbf{45.9_{\pm 2.7}}$	$\mathbf{44.5_{\pm 1.7}}$	$37.7_{\pm 5.7}$	$\mathbf{44.7_{\pm 7.0}}$	$30.5_{\pm 4.1}$	$45.8_{\pm 1.7}$	$\mathbf{32.0_{\pm 0.5}}$	$35.7_{\pm 2.5}$	$\mathbf{40.7_{\pm 2.9}}$	$29.7_{\pm 4.7}$	$\mathbf{44.8_{\pm 1.8}}$
S;X	$30.3_{\pm 1.8}$	$30.9_{\pm 1.5}$	$39.1_{\pm 4.5}$	$29.7_{\pm 4.7}$	$26.5_{\pm 4.2}$	$22.7_{\pm 2.2}$	$23.3_{\pm 1.7}$	$30.3_{\pm 2.6}$	$28.9_{\pm 1.3}$	$15.9_{\pm 2.9}$	$27.4_{\pm 2.2}$
C;G	$34.0_{\pm 2.4}$	$36.7_{\pm 1.5}$	$28.8_{\pm 1.8}$	$35.3_{\pm 6.3}$	$25.9_{\pm 2.6}$	$44.8_{\pm 3.5}$	$25.7_{\pm 2.4}$	$34.1_{\pm 3.5}$	$32.3_{\pm 4.7}$	$33.5_{\pm 5.1}$	$25.3_{\pm 2.8}$
C;E	$37.9_{\pm 5.9}$	$38.6_{\pm 1.3}$	$28.4_{\pm 7.1}$	$32.3_{\pm 3.7}$	$29.2_{\pm 2.4}$	$44.8_{\pm 2.2}$	$29.4_{\pm 1.0}$	$31.6_{\pm 6.6}$	$36.2_{\pm 4.6}$	$37.3_{\pm 6.7}$	$31.9_{\pm 2.6}$
C;B	$41.4_{\pm 1.4}$	$41.5_{\pm 1.7}$	$43.7_{\pm 2.7}$	$37.4_{\pm 7.0}$	$28.3_{\pm 2.3}$	$\mathbf{48.0_{\pm 4.2}}$	$26.6_{\pm 0.4}$	$34.6_{\pm 3.8}$	$33.4_{\pm 1.1}$	$\mathbf{38.5_{\pm 5.6}}$	$33.6_{\pm 5.1}$
C;X	$23.0_{\pm 3.4}$	$25.2_{\pm 1.1}$	$29.3_{\pm 6.9}$	$21.2_{\pm 3.5}$	$20.5_{\pm 2.5}$	$20.2_{\pm 2.6}$	$21.8_{\pm 1.9}$	$21.0_{\pm 2.8}$	$23.3_{\pm 1.8}$	$16.3_{\pm 1.9}$	$17.6_{\pm 4.6}$
H;G	$39.5_{\pm 4.2}$	$38.5_{\pm 1.8}$	$29.0_{\pm 4.6}$	$43.2_{\pm 3.6}$	$26.4_{\pm 3.9}$	$45.2_{\pm 3.5}$	$30.7_{\pm 1.5}$	$\mathbf{39.3_{\pm 4.0}}$	$37.8_{\pm 2.2}$	$36.2_{\pm 8.3}$	$39.6_{\pm 4.2}$
H;E	$36.6_{\pm 3.7}$	$39.9_{\pm 1.0}$	$34.5_{\pm 11.4}$	$31.8_{\pm 4.4}$	$\mathbf{36.2_{\pm 6.2}}$	$43.4_{\pm 2.5}$	$28.3_{\pm 1.5}$	$33.0_{\pm 1.4}$	$36.2_{\pm 3.1}$	$32.7_{\pm 6.6}$	$34.2_{\pm 6.9}$
H;B	$44.7_{\pm 2.9}$	$43.1_{\pm 1.7}$	$\mathbf{46.2_{\pm 2.8}}$	$\mathbf{44.6_{\pm 6.2}}$	$27.2_{\pm 3.2}$	$44.2_{\pm 2.0}$	$30.9_{\pm 2.1}$	$38.2_{\pm 2.4}$	$38.7_{\pm 2.1}$	$\mathbf{38.4_{\pm 6.2}}$	$36.7_{\pm 1.3}$
H;X	$30.0_{\pm 1.4}$	$30.7_{\pm 1.9}$	$40.0_{\pm 6.6}$	$27.7_{\pm 4.7}$	$24.8_{\pm 4.4}$	$21.1_{\pm 2.4}$	$24.5_{\pm 1.9}$	$33.2_{\pm 2.3}$	$29.5_{\pm 2.3}$	$17.8_{\pm 1.3}$	$16.9_{\pm 4.5}$

Table 5.6: Accuracy per question type over PororoQA using the frozen embeddings. Note that "*Abs*", etc. are introduced in Table 5.2 to identify the question types.

tional context. A similar reasoning could be applied to BERT as well: in fact, as shown in Table 5.5, when coupled with ST-VQA (row "S;B") it achieves the best result for this question type (45.4%), yet the mean value shown in Fig. 5.5.a is lowered due to the low average and high variance obtained by the other two architectures (with CoMem, i.e. row "C;B", it obtains 22.9% while with HME-VQA, i.e. row "H;B", 29.8%).

Similarly, the question type Obj_{1st} is best tackled with GloVe embeddings (Fig. 5.5.a reports 41.8% accuracy). In this case, around 78% of the questions are of the form "what am I holding", thus the long-term state provided by the LSTMs or by the Transformer encoder may be too complex to capture some of the information which, on the other hand, synergize well with the simplicity of GloVe.

PororoQA

Differently from EgoVQA, Fig. 5.5.b shows that BERT is the overall best choice when dealing with PororoQA, although there are situations in which GloVe and ELMo perform

	GloVe*	ELMo*	BERT*	XLM*	*avg*
ST-VQA	$35.7_{\pm 4.1}$(+0.0)	$34.0_{\pm 4.7}$(-0.9)	$34.9_{\pm 4.9}$(-1.2)	$25.1_{\pm 4.2}$(+1.1)	*-0.2*
CoMem	$33.9_{\pm 3.7}$(-0.2)	$32.4_{\pm 5.0}$(-1.2)	$33.1_{\pm 4.7}$(+1.5)	$22.3_{\pm 4.2}$(-3.4)	*-0.8*
HME-VQA	$35.6_{\pm 3.2}$(+0.2)	$34.9_{\pm 3.8}$(-0.6)	$\mathbf{36.4_{\pm 4.4}}$(+2.8)	$25.9_{\pm 4.0}$(-0.9)	*+0.4*
avg	*+0.0*	*-0.9*	*+1.0*	*-1.1*	*-0.2*

Table 5.7: Average accuracy (with absolute and average changes wrt Table 5.3) over EgoVQA after the finetuning of the embeddings.

comparably (*Stmt*, *Time*) or even better (*Loc*, *Reas*).

In this case, Table 5.6 is fundamental to detail some differences. HME-VQA coupled with ELMo performs better than all the other combinations for *Loc* questions: it achieves 36.2% accuracy (row "H;E"), while the second best is given by ST-VQA with BERT (row "S;B") which obtains 30.5% (hence, a 6% margin). Since the answers mainly differ due to nouns related to sceneries (e.g. "forest", "sea"), ELMo may be more effective at providing embeddings which cope better with these visual features. This proves extremely beneficial when coupled with the heterogeneous memory and the gating mechanisms exploited in HME-VQA, which help the model picking the correct association between the available multimodal features.

Obtaining "more than random" performance for *Stmt* questions is also noteworthy because these questions involve the contents of a *speech*. Given that GloVe, ELMo, and BERT obtain around 37% accuracy as shown in Fig. 5.5.b, several of these questions can be correctly answered to by only looking at the RGB frames, the question text, and the answer text. As an example of this, the possible answers for the question "what did Poby and his friend say" are permutations of "paper rock scissor": the models thus learn how to correctly answer by extracting spatio-temporal visual features which lead them to correctly identify the three hand gestures in the clip.

Another interesting detail can be seen in the *Caus* question type, which involves questions asking to identify an event which happened in relation to another one (e.g. "what happened when the egg broke?", "a green little dinosaur popped out"). The best result is obtained by BERT (Fig. 5.5.b reports around 42.5% accuracy), but XLM manages to shine as well (36.1%) obtaining a margin of at least 4% over ELMo (32.1%) and GloVe (30.3%). This is likely due to two facts: Transformer-based embeddings, and bidirectionality. The first point could be due to both the depth of the network and the attention mechanisms exploited in the Transformer, which are not used in GloVe nor in ELMo. The second point is supported by noticing that these questions are likely better understood when read both directions (the event described in the answer might have happened before the one described in the question, or vice versa), and by the fact that ELMo exhibits this property as well, leading it to be more accurate (around +2%) than GloVe.

5.4.5 Finetuning the embeddings

As a second set of experiments, we focus on the finetuning step of the embedding modules. This is usually performed because it helps the model gain a considerable

	GloVe*	ELMo*	BERT*	XLM*	*avg*
ST-VQA	$36.5_{\pm 0.5}$(-0.3)	$38.2_{\pm 1.0}$(+2.5)	$41.2_{\pm 1.0}$(+1.5)	$28.7_{\pm 0.8}$(+1.3)	*+1.2*
CoMem	$35.3_{\pm 0.8}$(+2.0)	$37.0_{\pm 1.5}$(+1.6)	$33.0_{\pm 7.2}$(-3.1)	$29.1_{\pm 0.4}$(+6.9)	*+1.9*
HME-VQA	$37.6_{\pm 1.0}$(+0.8)	$39.1_{\pm 1.9}$(+4.6)	$\mathbf{43.5_{\pm 2.0}}$(+4.3)	$29.3_{\pm 0.7}$(+1.5)	*+2.8*
avg	*+0.8*	*+2.9*	*+0.9*	*+3.2*	*+2.6*

Table 5.8: Average accuracy (with absolute and average changes wrt Table 5.4) over PororoQA after the finetuning of the embeddings.

M;E	Question type accuracy (%)										
	Abs	Act	Caus	Det	Loc	Met	Per	Reas	Stmt	Time	Y/N
S;G*	$37.2_{\pm 1.2}$	$42.5_{\pm 1.5}$	$27.4_{\pm 3.7}$	$35.7_{\pm 1.9}$	$24.7_{\pm 3.4}$	$45.4_{\pm 1.7}$	$\mathbf{34.6_{\pm 3.0}}$	$36.8_{\pm 5.7}$	$40.7_{\pm 2.1}$	$38.6_{\pm 3.5}$	$30.2_{\pm 2.0}$
S;E*	$39.3_{\pm 4.8}$	$41.1_{\pm 1.9}$	$35.8_{\pm 5.9}$	$38.8_{\pm 6.3}$	$30.8_{\pm 1.4}$	$46.2_{\pm 3.2}$	$34.1_{\pm 3.8}$	$35.0_{\pm 3.2}$	$44.4_{\pm 1.9}$	$34.0_{\pm 5.6}$	$42.7_{\pm 2.5}$
S;B*	$43.3_{\pm 4.4}$	$43.3_{\pm 1.7}$	$36.5_{\pm 7.7}$	$43.0_{\pm 7.1}$	$33.1_{\pm 4.7}$	$\mathbf{52.9_{\pm 3.2}}$	$33.9_{\pm 2.4}$	$40.7_{\pm 3.1}$	$43.0_{\pm 3.1}$	$31.9_{\pm 6.3}$	$\mathbf{48.6_{\pm 4.7}}$
S;X*	$28.8_{\pm 0.9}$	$34.8_{\pm 1.8}$	$37.4_{\pm 3.0}$	$30.2_{\pm 4.2}$	$23.9_{\pm 2.1}$	$20.9_{\pm 2.9}$	$25.0_{\pm 1.6}$	$34.4_{\pm 0.4}$	$28.4_{\pm 1.6}$	$22.3_{\pm 7.7}$	$17.1_{\pm 2.4}$
C;G*	$34.1_{\pm 2.9}$	$39.0_{\pm 2.1}$	$25.7_{\pm 2.3}$	$40.4_{\pm 5.7}$	$23.2_{\pm 4.7}$	$43.0_{\pm 3.4}$	$27.2_{\pm 2.5}$	$34.9_{\pm 2.8}$	$39.2_{\pm 4.6}$	$37.8_{\pm 6.4}$	$33.8_{\pm 2.8}$
C;E*	$38.0_{\pm 2.1}$	$38.2_{\pm 2.4}$	$26.0_{\pm 4.3}$	$36.2_{\pm 5.1}$	$32.5_{\pm 6.2}$	$41.1_{\pm 5.0}$	$32.0_{\pm 2.4}$	$35.5_{\pm 4.4}$	$42.0_{\pm 3.1}$	$38.9_{\pm 4.3}$	$37.7_{\pm 2.1}$
C;B*	$34.0_{\pm 7.5}$	$35.6_{\pm 8.6}$	$39.1_{\pm 3.4}$	$34.1_{\pm 5.9}$	$30.5_{\pm 7.0}$	$31.8_{\pm 10.2}$	$27.9_{\pm 6.7}$	$36.3_{\pm 8.2}$	$33.6_{\pm 5.9}$	$27.0_{\pm 7.1}$	$28.3_{\pm 14.8}$
C;X*	$33.4_{\pm 2.9}$	$33.5_{\pm 1.6}$	$\mathbf{45.2_{\pm 2.6}}$	$32.1_{\pm 1.6}$	$25.2_{\pm 3.7}$	$21.5_{\pm 1.6}$	$26.1_{\pm 2.3}$	$33.5_{\pm 2.2}$	$29.7_{\pm 1.3}$	$18.5_{\pm 2.5}$	$21.5_{\pm 1.5}$
H;G*	$38.3_{\pm 2.9}$	$37.8_{\pm 0.8}$	$27.8_{\pm 3.1}$	$43.1_{\pm 2.3}$	$26.0_{\pm 3.4}$	$41.1_{\pm 2.7}$	$33.7_{\pm 1.4}$	$42.4_{\pm 2.3}$	$39.3_{\pm 2.2}$	$38.1_{\pm 7.5}$	$38.4_{\pm 1.5}$
H;E*	$40.4_{\pm 4.1}$	$39.9_{\pm 2.4}$	$32.0_{\pm 4.1}$	$38.5_{\pm 3.8}$	$34.4_{\pm 3.5}$	$49.1_{\pm 4.2}$	$32.5_{\pm 1.8}$	$36.6_{\pm 2.6}$	$41.8_{\pm 1.3}$	$\mathbf{41.4_{\pm 3.5}}$	$39.7_{\pm 2.6}$
H;B*	$\mathbf{43.6_{\pm 3.6}}$	$\mathbf{43.6_{\pm 2.1}}$	$36.4_{\pm 1.4}$	$\mathbf{49.7_{\pm 5.5}}$	$\mathbf{34.9_{\pm 4.8}}$	$50.9_{\pm 3.8}$	$\mathbf{34.6_{\pm 2.3}}$	$\mathbf{45.1_{\pm 3.4}}$	$\mathbf{45.8_{\pm 2.1}}$	$33.8_{\pm 8.1}$	$46.4_{\pm 5.5}$
H;X*	$32.7_{\pm 0.6}$	$33.4_{\pm 1.3}$	$41.5_{\pm 5.3}$	$31.6_{\pm 3.8}$	$27.0_{\pm 2.7}$	$19.3_{\pm 1.9}$	$22.8_{\pm 2.9}$	$36.6_{\pm 1.4}$	$29.5_{\pm 1.8}$	$26.5_{\pm 4.6}$	$17.4_{\pm 1.8}$

Table 5.9: Accuracy per question type over PororoQA after the finetuning of the embeddings.

boost, since it helps rearranging the embedding space in a way to make the features more related to the task at hand. As an example, GloVe embeddings are finetuned in [73].

Instead of starting the training from scratch, we restart from the weights learned during the previous step. The procedure we follow can be described as such: first of all, we select the model with the best validation accuracy and use it to set the initial weights; then, we freeze all the components (but the embedding module) and perform the training for 20 epochs using a fixed learning rate of 5e-5. We repeat this procedure five times and we also use the same seed, i.e. when performing the i-th iteration of this procedure, we fix seed i and use the weights that were computed using seed i.

EgoVQA

From Table 5.7 it can be noted that, although the best result has improved, the finetuning procedure often does not help, e.g. with ELMo (on average, it loses around 0.9%). This is likely due to the dataset being too small to benefit from the finetuning, leading almost all the models to overfit. Yet, it can be seen that BERT benefits the most from this procedure (on average +1.0%) and, in particular, HME-VQA with BERT obtains a +2.8% improvement (+3.1% wrt the best result published in [72]).

PororoQA

Differently from EgoVQA, finetuning is particularly helpful and beneficial over PororoQA. Table 5.8 reports an average improvement of 2.6%, with a peak of +6.9% when

	GloVe	ELMo	BERT	XLM	*avg*
ST-VQA	**36.5**$_{\pm 4.7}$(+0.8)	33.7$_{\pm 5.0}$(-1.2)	**36.2**$_{\pm 6.9}$(+0.1)	25.2$_{\pm 3.1}$(+1.2)	*+0.2*
CoMem	32.9$_{\pm 3.2}$(-1.2)	33.8$_{\pm 4.3}$(+0.2)	31.2$_{\pm 4.5}$(-0.4)	24.0$_{\pm 3.1}$(-1.7)	*-0.8*
HME-VQA	34.5$_{\pm 3.4}$(-0.9)	35.1$_{\pm 4.0}$(-0.4)	35.0$_{\pm 5.5}$(+1.4)	27.2$_{\pm 3.9}$(+0.4)	*+0.1*
avg	*-0.4*	*-0.6*	*+0.3*	*-0.5*	*-0.2*

Table 5.10: Average accuracy (with absolute and average changes wrt Table 5.3) over EgoVQA after the adoption of the multi-task learning strategy (frozen embeddings).

	GloVe	ELMo	BERT	XLM	*avg*
ST-VQA	37.3$_{\pm 1.4}$(+0.5)	36.6$_{\pm 1.1}$(+0.9)	**40.6**$_{\pm 1.5}$(+0.9)	28.7$_{\pm 1.5}$(+1.3)	*+0.9*
CoMem	35.0$_{\pm 1.1}$(+1.7)	36.1$_{\pm 0.8}$(+0.7)	37.0$_{\pm 1.6}$(+0.9)	24.0$_{\pm 0.7}$(+1.8)	*+1.3*
HME-VQA	35.8$_{\pm 1.2}$(-1.0)	36.8$_{\pm 0.5}$(+2.3)	38.7$_{\pm 1.3}$(-0.5)	27.9$_{\pm 1.1}$(+0.1)	*+0.2*
avg	*+0.4*	*+1.3*	*+0.4*	*+1.1*	*+1.0*

Table 5.11: Average accuracy (with absolute and average changes wrt Table 5.4) over PororoQA after the adoption of the multi-task learning strategy (frozen embeddings).

finetuning XLM using CoMem. Table 5.9 shows a less varied situation than Table 5.6, with BERT being the overall best choice. Yet, some interesting results may be distilled from it.

First of all, after the finetuning step XLM achieves the best accuracy when dealing with the question type *Caus* (obtaining 45.2% when using CoMem). Although BERT obtains a lower average accuracy with respect to the previous performance (likely due to the learning rate being too high), BERT and XLM achieve 37.3% and 41.4% (Table 5.8), when averaging with respect to the architecture. Considering that GloVe and ELMo achieve 26.9% and 31.3%, this may confirm the previous hypothesis involving bidirectionality and network depth.

Secondly, ELMo receives on average a 5.3% improvement (from an average of 32.8% in Table 5.6 to 38.1% in Table 5.8, computed with respect to the three architectures) when tackling *Time* questions. Considering that these questions often involve reasoning about temporally-related events, the bidirectionality of ELMo coupled with the LSTM gating and memory capabilities may be the key point which helps understanding these relations. Although generally smaller, an improvement over *Time* questions is also observed with XLM and BERT.

Finally, HME-VQA coupled with BERT achieves both the best overall result (43.5% in Table 5.8) and also the best result over several question types. While this is partially due to BERT and its abilities to compute semantically rich embeddings, it surely confirms that HME-VQA is a powerful model able to capture multimodal cues which makes it great for VideoQA ([73]).

5.4.6 Adoption of multi-task learning strategy

As can be seen from the previous experiments, different word embedding techniques perform differently depending on the question type under analysis. This result is likely related to the different embedding techniques being able to capture some patterns in

M;E	Question type accuracy (%)										
	Abs	Act	Caus	Det	Loc	Met	Per	Reas	Stmt	Time	Y/N
S;G	$38.8_{\pm 2.8}$	$39.9_{\pm 2.8}$	$33.3_{\pm 5.7}$	$39.3_{\pm 3.6}$	$31.2_{\pm 3.9}$	$47.9_{\pm 3.0}$	$30.0_{\pm 1.6}$	$\mathbf{38.5_{\pm 2.5}}$	$42.4_{\pm 2.9}$	$\mathbf{39.5_{\pm 4.9}}$	$30.3_{\pm 5.1}$
S;E	$38.0_{\pm 1.0}$	$39.7_{\pm 2.0}$	$24.3_{\pm 4.0}$	$34.7_{\pm 0.7}$	$30.6_{\pm 4.4}$	$\mathbf{49.3_{\pm 2.6}}$	$31.0_{\pm 1.5}$	$34.9_{\pm 5.4}$	$41.6_{\pm 3.9}$	$38.5_{\pm 6.2}$	$39.6_{\pm 5.3}$
S;B	$43.6_{\pm 1.6}$	$\mathbf{42.2_{\pm 0.9}}$	$39.0_{\pm 7.1}$	$\mathbf{43.6_{\pm 5.2}}$	$30.3_{\pm 2.8}$	$46.3_{\pm 2.9}$	$\mathbf{35.3_{\pm 1.7}}$	$34.0_{\pm 4.5}$	$41.2_{\pm 3.3}$	$33.5_{\pm 4.0}$	$\mathbf{42.9_{\pm 4.0}}$
S;X	$28.9_{\pm 2.3}$	$30.5_{\pm 2.0}$	$\mathbf{45.0_{\pm 2.2}}$	$34.7_{\pm 3.5}$	$22.5_{\pm 4.6}$	$21.6_{\pm 1.6}$	$24.9_{\pm 1.2}$	$33.7_{\pm 2.1}$	$29.9_{\pm 2.1}$	$20.0_{\pm 4.7}$	$21.0_{\pm 1.5}$
C;G	$36.9_{\pm 2.6}$	$38.6_{\pm 1.9}$	$33.6_{\pm 5.6}$	$41.0_{\pm 4.7}$	$26.3_{\pm 2.4}$	$41.7_{\pm 3.2}$	$28.3_{\pm 3.9}$	$34.5_{\pm 4.3}$	$34.8_{\pm 3.2}$	$36.9_{\pm 4.1}$	$34.8_{\pm 3.3}$
C;E	$40.0_{\pm 2.1}$	$40.2_{\pm 1.9}$	$29.7_{\pm 7.9}$	$32.2_{\pm 2.7}$	$29.2_{\pm 3.9}$	$45.5_{\pm 2.2}$	$29.9_{\pm 1.8}$	$36.9_{\pm 3.1}$	$39.8_{\pm 3.5}$	$28.6_{\pm 7.3}$	$36.5_{\pm 4.3}$
C;B	$42.8_{\pm 1.9}$	$39.3_{\pm 2.2}$	$38.0_{\pm 8.8}$	$39.5_{\pm 4.4}$	$29.8_{\pm 3.9}$	$46.0_{\pm 2.8}$	$26.2_{\pm 2.4}$	$36.2_{\pm 1.8}$	$37.5_{\pm 3.5}$	$34.4_{\pm 5.0}$	$34.9_{\pm 3.8}$
C;X	$23.5_{\pm 4.2}$	$24.3_{\pm 3.8}$	$31.4_{\pm 2.6}$	$23.6_{\pm 4.2}$	$22.1_{\pm 2.3}$	$24.2_{\pm 2.6}$	$23.3_{\pm 1.5}$	$22.4_{\pm 4.0}$	$23.3_{\pm 2.4}$	$11.5_{\pm 2.5}$	$23.0_{\pm 4.6}$
H;G	$38.8_{\pm 2.5}$	$39.2_{\pm 0.9}$	$36.7_{\pm 4.7}$	$41.6_{\pm 4.6}$	$27.6_{\pm 2.5}$	$45.4_{\pm 3.6}$	$29.1_{\pm 2.4}$	$37.7_{\pm 6.1}$	$41.8_{\pm 1.3}$	$23.9_{\pm 5.6}$	$32.7_{\pm 3.4}$
H;E	$38.9_{\pm 2.1}$	$41.2_{\pm 0.9}$	$26.2_{\pm 6.2}$	$36.1_{\pm 5.0}$	$\mathbf{34.3_{\pm 4.2}}$	$44.9_{\pm 4.1}$	$32.1_{\pm 1.1}$	$30.2_{\pm 4.6}$	$42.2_{\pm 3.6}$	$35.0_{\pm 4.7}$	$40.0_{\pm 5.7}$
H;B	$\mathbf{44.4_{\pm 1.2}}$	$42.0_{\pm 3.1}$	$41.5_{\pm 8.3}$	$40.3_{\pm 2.0}$	$26.4_{\pm 1.0}$	$45.7_{\pm 4.7}$	$32.1_{\pm 2.9}$	$36.9_{\pm 4.3}$	$\mathbf{43.7_{\pm 4.9}}$	$31.0_{\pm 3.8}$	$38.4_{\pm 4.6}$
H;X	$29.5_{\pm 2.5}$	$32.3_{\pm 1.3}$	$43.6_{\pm 7.7}$	$26.2_{\pm 1.7}$	$19.7_{\pm 2.5}$	$22.4_{\pm 1.0}$	$25.9_{\pm 2.2}$	$34.8_{\pm 2.8}$	$31.1_{\pm 2.3}$	$20.0_{\pm 6.4}$	$25.2_{\pm 1.9}$

Table 5.12: Accuracy per question type over PororoQA after the adoption of the multi-task learning strategy (frozen embeddings).

the question which depend on the type and are helpful to localize the answer within the video. To prove this surmise, we propose to adopt a multi-task learning strategy (detailed in Sec. 5.3.1), and show that it helps the models achieve a better generalization.

EgoVQA

Table 5.10 shows that HME-VQA coupled with BERT is the combination which benefits the most from the adoption of the multi-task strategy, gaining on average 1.4% accuracy. Yet, several of the other combinations gain only marginal improvements or even obtain a lower accuracy. This is likely due to the models overfitting even more than before, due to the added parameters.

Nonetheless, there are also other combinations which benefit from the added parameters as well. In particular, it can be noted that ST-VQA synergizes the best with the proposed technique, since it improves its performance when coupled with GloVe (+0.8%) and XLM (+1.2%) embeddings. This may be due to its simplicity and lower amount of parameters with respect to CoMem and HME-VQA.

To understand whether the difference in performance before and after the addition of the proposed multi-task learning strategy is significant, we use the Almost Stochastic Order (ASO) test ([49, 60]), as implemented by [277]. This test operates on the cumulative distribution function of the two models (before and after training with the proposed strategy) and estimates the amount of violation of the stochastic order. It formulates the following null hypothesis: $H_0 : \epsilon_{\min} \geq \tau$, which can be interpreted as the standard training being stochastically dominant in more cases than the training performed with the proposed strategy. To reject this hypothesis, $\epsilon_{\min} < \tau$ where $\tau = 0.5$. Using ASO with a confidence level $\alpha = 0.05$ it was possible to confirm some of the results we observed. In particular we found that, based on five random seeds, ST-VQA with ELMo, CoMem paired with GloVe or XLM, and HME-VQA paired with GloVe are stochastically dominant (in particular, $\epsilon_{\min} \geq 0.80$) if trained without the proposed strategy; in the cases of HME-VQA with ELMo or XLM, CoMem with ELMo or BERT, and ST-VQA with BERT the $\epsilon_{\min}$ is close to the threshold τ, so the difference is not as significant. In the other cases, the addition of the multi-task learning strategy leads to stochastically dominant solutions, with $\epsilon_{\min} \leq 0.40$ in most cases.

PororoQA

As can be seen in Table 5.11, almost all the different combinations of architectures and embeddings benefit from the adoption of the multi-task learning strategy: overall, if we compare to the results obtained before the finetuning of the embeddings (Table 5.4, changes are reported in Tab. 5.11), the improvement obtained by adopting the proposed strategy amounts to around +1.0% with a peak of +2.3% when using HME-VQA with ELMo. This shows that such simple addition helps the models both generalize better and understand what they should focus on based on the type of the question.

To ease the comparison of Tables 5.6 and 5.12, we propose in Fig. 5.6 a simplified view of the results per question type, where we visualize the average accuracy obtained by the embedding techniques (mean values with respect to the three architectures) before and after the adoption of the multi-task learning strategy. While it shows an overall improvement, such a figure also shows there are situations in which the proposed technique is beneficial or not based on the embedding technique used. As an example, for *Caus* questions, GloVe and XLM benefit greatly from the additional task, whereas BERT and ELMo do not. More in detail, Table 5.12 shows that a considerable boost (+9.9%) is obtained by ST-VQA coupled with ELMo when dealing with *Time* questions, where the accuracy goes from 28.6% (in Table 5.6, row "S;E") to 38.5%. Since a similar improvement was also obtained when finetuning ELMo for this question type, this further strengthens the hypothesis previously formulated.

As in the previous case, we use ASO to determine the significance of the results. With a confidence level $\alpha = 0.05$, we found that the addition of the proposed multi-task learning strategy leads to solutions which are stochastically dominant over the model trained without the proposed strategy in most of the cases (with $\epsilon_{\min} < 0.40$). In the case of HME-VQA, the addition of the proposed strategy leads to a stochastically dominant solution only when paired with ELMo ($\epsilon_{\min} < 0.10$) whereas, according to the statistical test, the model trained without the proposed strategy is either stochastically dominant (with GloVe and BERT, $\epsilon_{\min} \geq 0.85$) or the same as the model trained with the proposed strategy (with XLM, $\epsilon_{\min} = 0.5$).

5.4.7 Embeddings finetune after the multi-task learning

As we did previously, we start from the weights obtained during the training with the multi-task learning strategy and proceed with the finetuning of the embeddings alone. Overall, a greatly positive outcome is achieved for PororoQA (Table 5.14), whereas over EgoVQA it does not help at all.

PororoQA

Table 5.14 reports on average a +2.9% improvement, with even higher peaks when using CoMem (+5.7%). Also in this case we propose a simplified view of the comparison between Tables 5.12 and 5.14 in Figure 5.7, where we visualize the accuracy obtained before and after finetuning. From the Figure it can be seen that BERT and ELMo always benefit from the finetuning procedure. In particular, from Table 5.14 it can be seen that the average improvement amounts to 1.8% for ELMo and 3.9% for BERT. It

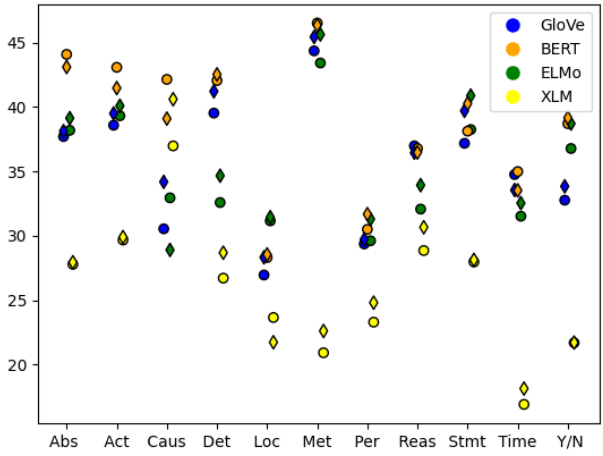


Figure 5.6: Average accuracy over PororoQA, averaged with respect to the three architectures, before (○) and after (◊) the adoption of the proposed multi-task learning strategy. Best viewed in color.

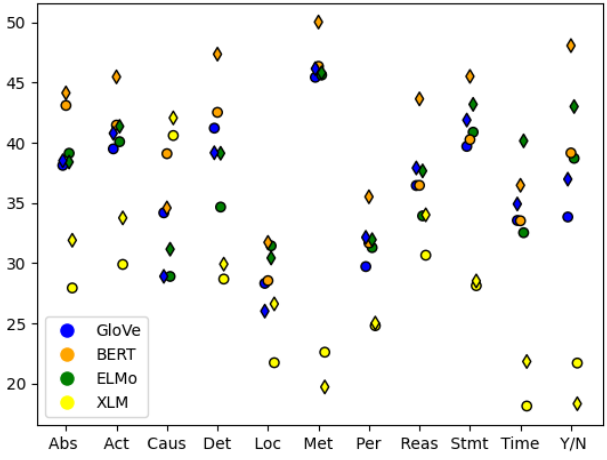


Figure 5.7: Accuracy over PororoQA obtained by the embedding techniques (average wrt models trained with the proposed multi-task learning strategy) before (○) and after (◊) finetuning. Best viewed in color.

	Embedding technique				
Architecture	GloVe	ELMo	BERT	XLM	*avg*
ST-VQA	0.005	0.039	0.015	0.026	0.021
CoMem	0.064	0.101	0.074	0.095	0.083
HME-VQA	0.050	0.085	0.062	0.081	0.069
avg	0.040	0.075	0.050	0.067	-

Table 5.13: The average time (in seconds) spent per sample at inference time for each of the 12 combinations considered in this Chapter.

follows that, generally speaking, it is a wise choice to finetune the embeddings, especially when there is a decent amount of available data.

5.4.8 About inference times

In this section, we analyze the time taken to predict the answer during the inference phase. In particular, the total time required by the pipeline analyzed by isolating the feature extraction of the video from all the other operations. This is done because the visual feature extraction is performed in the same way for all the considered solutions. In fact, all of them, as previously described in Sec. 5.3, use VGG and C3D which, on average, take less than 150 ms per video clip. Therefore, the time taken by this step can be removed from the total time in order to make it clear the overhead taken by the specific architecture or word embedding techniques. Conversely, the extraction of the textual features is tightly linked to the word embedding technique used. In Table 5.13 we report the average time taken by all the combinations of overall model (ST-VQA, CoMem, or HME-VQA) and word embedding technique considered in this Chapter. According to this analysis, ST-VQA combined with GloVe represents the fastest solution taking only 5 ms on average. In particular, GloVe represents the fastest word embedding approach since it only needs to map tokens to vectors through a table, whereas BERT, ELMo, and XLM have additional layers which slow down the process, leading respectively to 50, 75, and 67 ms on average. Moreover, since it is the simplest architecture considered in this Chapter, ST-VQA is also the fastest among the three (on average, 21 ms compared to 83 and 69 ms taken by CoMem and HME-VQA).

5.4.9 Take-home messages

Word embeddings. Each of the analyzed embedding techniques deals better with specific question types, likely implying questions have characteristics which are encoded differently (and possibly ignored) by each technique. Over EgoVQA, ELMo works better when identifying an actor (Who_{3rd}), and GloVe is effective when identifying objects (Obj_{1st}); over PororoQA, ELMo performs significantly better than GloVe, XLM, and BERT when identifying locations (Loc), and the synergy between the bidirectionality and the Transformer-based encoder, used by XLM and BERT, is beneficial when guessing which event happened in relation to another one ($Caus$).

Importance of the embedding choice. In relation to the previous message, we thoroughly showed that the choice of the embedding technique to use should take into

	GloVe*	ELMo*	BERT*	XLM*	*avg*
ST-VQA	$37.8_{\pm 1.2}(+0.5)$	$38.8_{\pm 0.9}(+2.2)$	$\mathbf{42.7}_{\pm \mathbf{1.4}}(+2.1)$	$30.0_{\pm 0.6}(+1.3)$	*+1.5*
CoMem	$35.6_{\pm 0.6}(+0.6)$	$37.8_{\pm 1.2}(+1.7)$	$42.7_{\pm 1.8}(+5.7)$	$28.7_{\pm 1.2}(+4.7)$	*+3.2*
HME-VQA	$37.3_{\pm 0.9}(+1.5)$	$38.2_{\pm 1.2}(+1.4)$	$42.6_{\pm 1.1}(+3.9)$	$28.9_{\pm 0.6}(+1.0)$	*+1.9*
avg	*+0.9*	*+1.8*	*+3.9*	*+2.3*	*+2.9*

Table 5.14: Average accuracy (with absolute and average changes wrt Table 5.11) over PororoQA after the adoption of the multi-task learning strategy and the finetuning of the embeddings.

account which question type (and the properties of its questions) is the most prevalent in the considered dataset.

Bidirectionality. We provide evidence showing that bidirectionality is convenient when both Q and A are rich and complex sentences (e.g. *Caus*, *Time* questions). Although for the latter it becomes clearer when finetuning, for the former it is noticeable also when using the frozen embeddings.

Finetuning. Although the improvements due to finetuning are harder to see with EgoVQA due to its smaller size, it is diaphanous for PororoQA: finetuning helps rearranging the embedding space, making it easier for the models to understand and link the textual and the visual concepts.

Multi-task learning. Question types raise the possibility to perform finer-grained analysis, but they can also be exploited to achieve improved generalization. We show this is possible by proposing a multi-task learning strategy which takes question types into account.

5.5 Conclusion

VideoQA has recently seen a surge of interest thanks to the release of several rich and public datasets. In VideoQA, to provide a meaningful answer, the model needs to understand both the visual and the textual content. Given the multitude of word embedding techniques and considering that the computed representations influence the final performance of the VideoQA model, we explore the use of several of them on two public datasets: EgoVQA and PororoQA. We find that the embeddings computed by BERT are the best overall solution, but we also find that depending on the question type different embeddings should be preferred.

Moreover, we showed this result can be further exploited by introducing a multi-task learning approach where the models are also asked to classify the given questions: a simple yet effective technique which greatly helps the overall performance of the considered solutions.

Finally, we show that more accurate predictions can be obtained by finetuning the embeddings, both with and without the proposed multi-task learning strategy. BERT is the technique benefiting the most from it, but there are situations in which the improvement can be substantially large when taking into account specific question types, e.g. the synergy between ELMo and *Time* questions. At the end of the experimental section we also collect some "take-home messages" (Sec. 5.4.9) where we summarize the

main results observed in this Chapter.

As a future work, several other word embedding techniques can be tested, such as DistilBERT ([242]) and RoBERTa ([176]). Moreover, we focused on EgoVQA and PororoQA, but these results should help obtaining improved performance in several other datasets, such as TGIF-QA ([120]) or MovieQA ([273]), where it is possible to define the type of the questions. In particular, automatically identifying the type of the question may be an interesting research direction. The types may be defined by a rule-based system (e.g. inspired by the "five Ws"), or by using clustering techniques to automatically discover clusters of semantically related questions. Furthermore, in our multi-task learning approach we focused on a single auxiliary task designed on the concept of question type, but additional tasks may be used to extend it and help the model extracting more general features. Finally, the VideoQA community is mostly focusing on defining new methods to achieve better performance. Nonetheless, predicting the correct answer with a lower time delay may have important consequences on several applications, therefore it may become an interesting research direction.

6

Relevance-based Margin for Contrastively-trained Video Retrieval Models

6.1 Introduction

With the rapid growth of digital media shared on the web it becomes increasingly important for real-world applications to offer flexible, user friendly modalities to access media content at scale. Google video search for example, translates a natural language query into a ranked list of content-related videos from the web. Natural free form, unrestricted language enables a user to express the fine-grained details in an articulated query, and each user can do so with its own expressivity. Thus, a same retrieval response can be triggered with syntactically different but semantically coherent queries. This poses significant challenges to the current state of the art in cross-modal retrieval research.

Recent approaches which deal with cross-modal video retrieval aim at learning a joint embedding space [26, 40, 57, 288] by means of contrastive losses [93, 247, 193, 205], which put the associations available in the dataset (e.g. a video and its natural language description) as close as possible while enforcing a separation margin to all the other items (see lower left of Fig. 6.1). During inference, the ranking list for a given query is produced by computing similarity scores with respect to all the items by means of, e.g. the dot product or the cosine similarity. By measuring the performance of the video retrieval system with rank-unaware metrics, such as recall rates, increasingly better solutions to this problem were proposed. In fact, contrastive losses synergize well with recall rates, given how they maximize the similarity of the associated items. But during training they do not make any distinction between items which are *highly relevant* and items which are only *partially* or *completely irrelevant* to a given query. For example, if a query is about 'how to cook a pizza', then videos which depict how to 'bake a pizza', 'cook pasta', or 'knead dough' are all treated the same way, although they can be more

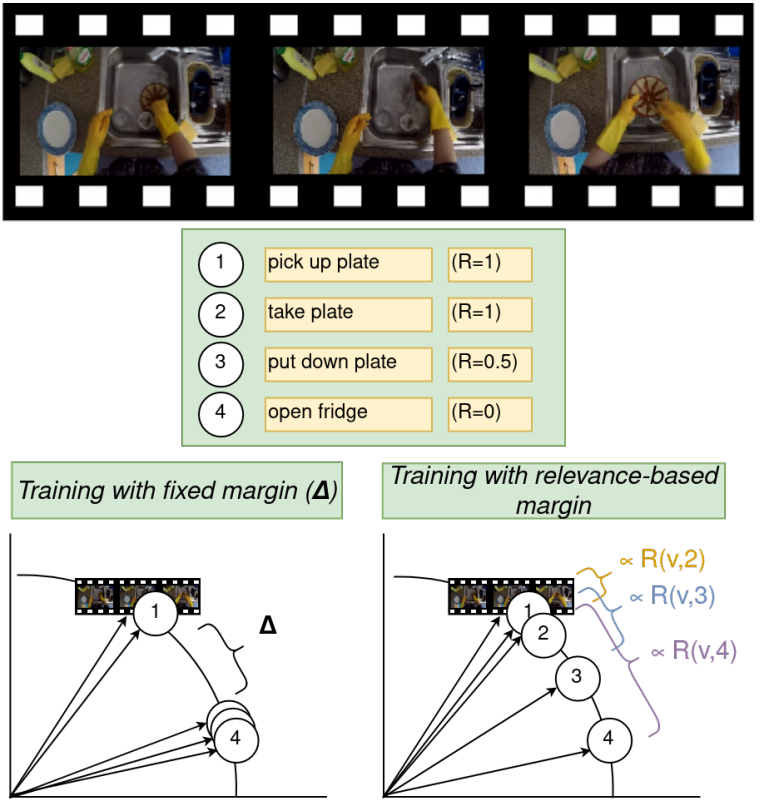


Figure 6.1: Training a model for text-video retrieval by employing a contrastive loss which uses a fixed margin Δ (lower left) treats semantically equivalent descriptions which do not appear as groundtruth pairs in the dataset as equally irrelevant. We propose to move away from such a paradigm and adopt a relevance-based margin (lower right), i.e. a margin which is proportional to the relevance $\mathcal{R}$ (see Sec. 6.3.1).

or less semantically close to the query. Furthermore, one of the reasons which limited the usage of rank-aware metrics in video retrieval consists in visual-language datasets only providing the visual contents and textual annotations (obtained manually [310, 343] or automatically [195]). Due to the absence of relevance grades, rank-aware metrics (e.g. nDCG) are difficult to adopt. Recently, this problem was partially alleviated by the introduction of a relevance function [45] which, to avoid a costly manual annotation step, is defined in terms of the captions already available in the dataset.

To give the model awareness of the semantical differences between items and queries during training, we free the margin from its stillness. Several solutions for non-fixed margins were proposed in previous literature, such as using multiple margins (e.g. [33]) or adaptive solutions. In particular, [248] implemented a schedule for the margin value which gradually incorporates inter-category correlations and information about the structure of the embedding space. Recently, for video retrieval [96] proposed an adaptive margin proportional to the similarity of item and query as computed by multiple

models. Differently from them, we propose to inject semantic knowledge into the training process by means of a relevance-based margin. To do so, we leverage the relevance function detailed in [45], so that the margin is proportional to how relevant the item is to the query, as illustrated in Fig. 6.1. By doing so, we effectively discard one hyperparameter to tune. Moreover, even by performing an expensive search for it, the results are still suboptimal when compared to the proposed relevance-based margin. We give empirical evidence that the proposed technique makes it possible to easily improve the quality of the ranking lists, measured through Normalized Discounted Cumulative Gain (nDCG) and Mean Average Precision (mAP). We use three different and increasingly more complex models (MME from [299], JPoSE [299], and HGR [26]) on two datasets (EPIC-Kitchens-100 [45] and YouCook2 [343]). Furthermore, we perform several ablations to study how it interacts with multiple video modalities (motion, appearance, audio) and with both cross-modal and within-modal losses.

We organize the Chapter as follows. In Section 6.2 we review related works, including vision and language tasks, main techniques and losses used to deal with text-video retrieval, and optimization of retrieval metrics such as the nDCG. Then, we formally describe the proposed technique in Sec. 6.3, in terms of the relevance function and how we apply it to a typical contrastive loss setting. In Sec. 6.4 we perform multiple experiments to prove the strength of the relevance-based margin. Finally, in Sec. 6.6 we conclude the Chapter.

6.2 Related works

Vision and Language. In recent years, deep learning brought several advancements in multiple tasks dealing with vision and language, such as question answering [4, 5, 111, 137], retrieval [150, 335, 57, 26], and captioning [254, 59, 152, 157]. Given that vast amounts of data can be scraped from the web, many works perform a joint vision and language pretraining [165, 31, 270, 342] by optimizing vision-text proxy tasks. Recently, a line of research uses natural language supervision such as captioning [52] or alignment [123] objectives to pretrain visual models. While in both cases they achieve competitive and state-of-the-art results on downstream tasks, these methods are data hungry and expensive to train, making them impractical from a computational point of view.

Text-Video Retrieval. Multiple techniques were proposed to learn a representation for the input data while capturing multimodal interactions. [175, 288, 76] explore multimodal fusion techniques to fuse all the information extracted from a video using multiple pretrained 'experts'. While these methods focus on the addition of video-side information, a supervisory signal can also be obtained by looking with more detail at the text. [26] create a semantic role graph of the caption and aligns to each node a learned representation of the clip-level descriptor. [299] extract verbs and nouns from the caption and uses them to learn Part-of-Speech-specific embedding spaces. [212] introduce a generative cross-captioning task, using the batched videos as a support set. Recently [40] distil information from multiple pretrained text experts. A different trend involves heavy pretraining steps [61, 151, 12, 173], followed by finetuning for downstream tasks. Moreover, the addition of image-text datasets as part of the pretraining step, showed significant improvements when dealing with video-related tasks [151, 12]. While these

methods achieve impressive results, they rely heavily on the data, are expensive to train, and are not designed for the nature of the problem.

Due to the unavailability of groundtruth relevance values which can inform about the optimal ranking list to a given query, the video retrieval community focused on rank-unaware metrics such as the recall rates or the median rank. Contrastive losses greatly improve these metrics since they reduce the distance between the visual descriptor and the linguistic one and thus increase its similarity, making it possible to retrieve it before the negative descriptors. But multiple descriptions can be equally or partially relevant for the same video (and vice versa), thus more complex and rich metrics, such as the nDCG, are needed to accurately evaluate a retrieval system [298]. To do so, a way to determine how relevant an item is to a query must be available. To avoid the need for manual and costly annotation, [45] proposes to use a relevance function defined in terms of the noun and verb classes present in the caption (more details in Sec. 6.3.1).

Learning a joint embedding space. Common approaches for text-video retrieval learn a joint embedding space by means of a contrastive loss [93, 247] which, during training, puts semantically similar items (e.g. a video and a caption describing its contents) closer in the embedding space, while dissimilar items are pushed away. While groundtruth associations (i.e. positive pairs, such as a video and its caption) are known from the dataset, the negative examples (such as a different video) have to be sampled, or 'mined', given that the amount of possible tuples scales exponentially with the dataset size, e.g. cubically with triplets. Multiple techniques have been proposed including: offline mining, which randomly samples a fixed number of tuples and repeats the process multiple times during training; online mining, which uses the negatives inside the mini-batch by considering all the non-groundtruth pairs, or only hard [100, 313] or semi-hard negatives [247]. Recent research also found relevant signal while mining positive samples, e.g. easy [314] or hard positives [100]. In this Chapter, we focus on triplets as they are a popular margin-based contrastive loss, but it can be extended to other techniques, e.g. to quadruplets [29]. Moreover, we experiment with two different and opposite techniques: offline mining with random sampling and online mining with hard negatives, and show the advantages of the relevance-based margin in both cases.

Margin in contrastive losses. Most of the approaches involving contrastive losses are based on maximum-margin losses (e.g. [93]). Although the margin is usually fixed, variable or adaptive solutions for it have been explored in different fields. For person re-identification, [33] suggest using two different (but fixed) margins for inter- and intra-class constraints, whereas [336] propose to monotonically increase the margin during the training process. [109] use a 'soft margin' to improve recommender systems, that is they remove the fixed margin and use (a soft version of) the distance between positive and negative pairs as the loss. [159] augment the bidirectional contrastive loss by also summing the margin to the loss objective, to optimize it during the training process. For text-image retrieval, [248] propose a scheduled adaptive margin which starts from a fixed value and gradually changes during the training process both to integrate inter-category similarity-based correlations and to preserve the category clusters formed during the initial phases of the training. Recently, for cross-modal video retrieval [96] proposed an adaptive margin proportional to the similarity of the representations computed for the negative pair, both in terms of 'static' (pretrained, frozen) models, which provide initial supervision, and 'dynamic' (trained with the task) models, which provide supervision in

later stages of the training. Differently from all these works, we propose a margin which is proportional to the relevance value of the queries involved in the triplet, effectively using the semantic knowledge during training.

Optimization of nDCG. Considering that visual-textual datasets usually lack relevance grades, rank-unaware metrics are one of the preferred ways to measure progress in the video retrieval community. Yet given a video, multiple captions can be used to describe its contents. To capture the difference in the ranking list when binary relevance (i.e. a caption is either relevant or irrelevant to a video) is considered, mAP is preferred to the recall rates. Furthermore, finer-grained relevance grades could be also available (i.e. a caption can be relevant to a video to some degree), in which case the DCG (or its normalized version, the nDCG) is chosen. But, optimizing these metrics during training clashes with gradient-based optimization methods because ranks are not differentiable with respect to the learnable parameters, e.g. the nDCG of a list of items to a given query is normalized using the optimal ranking list, which is computed by *sorting* with respect to the relevance values.

Surrogate losses are used to partially address this problem, which can be categorized into: pointwise (e.g. regression loss [39]), which compare predicted and optimal rank of one item at a time; pairwise (e.g. RankNet [21]), which deal with pairs of items and relative ordering; listwise approaches (e.g. LambdaRank [20]), which work on full list of items. Note that the triplet loss [247] can be seen as a 'triplet-wise' surrogate loss. Since these surrogate losses are loosely connected to downstream metrics, there is also an active research field which directly optimizes retrieval metrics by deriving a relaxation of the sorting operator which has well-defined gradients, e.g. [90, 42, 15].

Considering its widespread usage for video retrieval, we consider the triplet loss an optimal candidate for our relevance-based margin, and show it can lead to higher quality ranking lists.

6.3 Relevance-based margin

In Sec. 6.3.1 we define the relevance function $\mathcal{R}$ and the metrics used during evaluation. In Sec. 6.3.2 we describe how we change the margin in the contrastive loss to make it dependent on $\mathcal{R}$. Finally, Sec. 6.3.3 details the three methods on which we test our technique.

6.3.1 Semantic classes and relevance

Given a video clip, multiple natural language descriptions may fully capture its visual contents, and vice versa. Hence, if a user looks for videos about 'cooking a pizza', an intelligent video retrieval system should retrieve all the videos which show how to cook a pizza, and show them all before (i.e. rank them higher than) those that show the baking of a 'focaccia'. Similarly, videos about 'fried potatoes' should be ranked even lower, given how dissimilar they are when compared to the user query. As a consequence, the automatic evaluation of the quality of a ranking list requires a function which considers 'focaccia' more relevant than 'potatoes' when compared with 'pizza', as well as the cooking technique ('bake' versus 'fry'). To avoid the need for costly manual annotation which requires human assessments using a predefined set of grades, [45] introduces a

relevance function $\mathcal{R}$ defined as:

$$\mathcal{R}(x_i, x_j) = \frac{1}{2}\left(\frac{|x_i^V \cap x_j^V|}{|x_i^V \cup x_j^V|} + \frac{|x_i^N \cap x_j^N|}{|x_i^N \cup x_j^N|}\right) \tag{6.1}$$

where x_k^V and x_k^N denote the sets of verb and noun classes found in the k-th caption. This can be extended to videos by considering the associated description. By defining the relevance as in Eq. 6.1, x_i is highly relevant to x_j if they share similar noun and verb classes. We refer to 'classes' because we do not want to consider synonyms (e.g. 'pick up' and 'take', or 'drop' and 'put down') as different items which need to be separated, hence each class will contain tokens with a similar meaning. In some datasets, this class knowledge may be already available, but several other datasets do not provide it. To automatically compute them, a pipeline made of a PoS-tagger (e.g. with spaCy), followed by WordNet [198] and the Lesk algorithm [155] can be used, as in [298].

To evaluate a video retrieval system, we use two metrics which are commonly used in Information Retrieval, which are the Mean Average Precision (mAP [10]) and the Normalized Discounted Cumulative Gain (nDCG [122]), as recently proposed in [298]. The mAP is defined as the mean of the Average Precision (AP) with respect to all the queries. For a given query q, AP can be defined as:

$$AP(q) = \frac{\sum_{k=1}^{N} P(k) \cdot r(k)}{N_r} \tag{6.2}$$

where N is the number of items (both relevant and irrelevant) in the ranking list, $P(k)$ is the Precision at k [10], $r(k)$ is an indicator function which tells whether the k-th item is relevant or not, and N_r is the total number of relevant items. The mAP allows to grasp with a single number the area under the Precision-Recall curve. But this metric requires binary relevance values, thereby requiring the introduction of a threshold below which items are considered irrelevant (and relevant otherwise). For mAP, we consider k to be relevant to q only when $\mathcal{R}(q, x_k) = 1$ as is done in [45] (hence, for mAP $N_r = |\{x_i \,|\, \mathcal{R}(q, x_i) = 1\}|$). On the other hand, nDCG makes use of non-binary relevance values, allowing it to grasp finer details (and errors) of the ranking list. Given a query q and a list of items $K = \{x_i\}$, it is defined as

$$nDCG(q, K) = \frac{DCG(q, K)}{IDCG(q, K)} \tag{6.3}$$

where IDCG is the optimal DCG value obtained when the ranking list follows a descending order of relevance values. We define DCG as in [122, 45]:

$$DCG(q, K) = \sum_{k=1}^{N_r} \frac{\mathcal{R}(q, x_k)}{\log_2(k+1)} \tag{6.4}$$

where x_k is the k-th item in the list K, and we only consider the first N_r items in the ranking list. Note that $N_r = |\{x_i \,|\, \mathcal{R}(q, x_i) > 0\}|$.

6.3.2 Contrastive loss with relevance-based margin

To learn a joint text-video embedding space, various contrastive (or ranking) losses have been proposed (see Sec. 6.2). In this Chapter we consider a contrastive term based on the triplet loss defined as:

$$\mathcal{L} = [m + s(a, n) - s(a, p)]_+ \tag{6.5}$$

where $[\cdot]_+ = \max(0, \cdot)$, m is interpreted as a separation margin, $s(\cdot, \cdot)$ is a similarity metric (e.g. cosine similarity), whereas a, n, and p represent respectively the embedding of the *a*nchor, *n*egative, and *p*ositive item. Eq. 6.5 provides a positive loss when the margin m between the positive pair (a, p) and the negative one (a, n) is violated, i.e. $s(a, p) - s(a, n) < m$. The loss may be cross-modal, i.e. n, p from one modality (e.g. video) and a from the opposite one (e.g. text), or within-modal, i.e. a, p, n are all from the same modality. Furthermore, the optimal m is not known beforehand and should be treated as an hyper-parameter which can affect the performance. Thus, it should be tuned on the validation set.

During training, all the items which are not from the positive pair (a, p) are pushed away until they are separated by a margin of m, as shown in Fig. 6.1. Although effective and widely used in the literature, Eq. 6.5 ignores that multiple items may be completely or partially relevant to the same query, and treats all the items which are not from the groundtruth pair as equally irrelevant. Thus the retrieval system might not be able to distinguish between the many relevance levels which can exist between an item and a query.

To address this, we propose a relevance-based margin instead of a fixed margin. In this Chapter, we aim at defining m in terms of the relevance function $\mathcal{R}$. In particular, we update Eq. 6.5 as follows:

$$\mathcal{L} = [\Delta_{a,p,n} + s(a, n) - s(a, p)]_+ \tag{6.6}$$

where:

$$\begin{aligned} \Delta_{a,p,n} &= R(a, p) - R(a, n) \\ &= 1 - R(a, n) \end{aligned} \tag{6.7}$$

since we consider the groundtruth pair to be maximally relevant, i.e. $R(a, p) = 1$. The relevance-based margin keeps $\mathcal{L}$ positive until $s(a, p)$ and $s(a, n)$ are separated by a margin which is proportional to their relevance values, thus separating irrelevant items more than those which have a positive relevance. This is illustrated in Fig. 6.1 on the lower right. Note that this term is not bound to the network state and can thus be applied both to offline and online mining techniques.

6.3.3 Methods

Given a dataset $D = \{(v_i, q_i)\}$ of video-caption pairs, we strive to learn optimal weights for two embedding functions $f : \mathbb{R}^{f_v} \rightarrow \mathbb{R}^d$ and $g : \mathbb{R}^{f_q} \rightarrow \mathbb{R}^d$ such that $f(v_i)$ and $g(q_i)$ are close in the d-dimensional joint embedding space. Here f_v and f_q represent the dimension of the video and caption descriptors. To parameterize f and g we con-

sider the following methods: **MME** is a baseline from [299] which learns one embedding function for each of the two modalities, video and text. In **JPoSE [299]**, the captions are processed in order to obtain a single sentence-level descriptor and multiple descriptors restricted to specific Part-of-Speech (PoS) tags, e.g. nouns and verbs. Then, two functions are learned for each of these embedding spaces using both cross-modal and intra-modal contrastive terms for the sentence-level, as well as for the PoS-level. **HGR [26]** structures the learning at multiple levels (global event, local actions, and local entities) which are obtained by computing a semantic role graph for each of the captions. Then a graph convolutional network is used to learn compositional semantics of the caption based on the local components, i.e. full sentence, verbs, and noun phrases.

We choose these three methods because they provide incrementally structured approaches to deal with video and language data, starting from a simpler MLP-based network to a graph-based approach. Moreover, JPoSE represents the state-of-the-art for EPIC-Kitchens-100 (measured through nDCG and mAP), which is the main dataset under consideration. Finally, by selecting them we can validate our approach on both offline (MME and JPoSE) and online (HGR) mining techniques. We thus proceed to show the generality and effectiveness of the proposed relevance-based margin by empirically validating on two different datasets.

6.4 Experiments

After the introduction of the experimental setting in Sec. 6.4.1, we show in Sec. 6.4.2 how the proposed relevance-based margin helps to achieve better nDCG and mAP on EPIC-Kitchens-100 and YouCook2. Then, in Sec. 6.4.3 we perform several ablation studies. First we show that even by carefully tuning the fixed margin, the proposed technique consistently achieves better results without the need to tune it. Secondly, we also evaluate its robustness by ablating the loss function and the modalities used in JPoSE. Finally in Sec. 6.4.4 we analyze the distribution of the margin values during training and some video-to-text examples from the testing set.

6.4.1 Experimental setting

Datasets. We focus our experimental setting on two challenging video and language datasets: the recently released EPIC-Kitchens-100 [45] and YouCook2 [343]. For the retrieval challenge, the former comprises 67217 egocentric clips for training and 9668 for evaluation. It is also the largest dataset for video retrieval in the egocentric setting. Moreover, it also provides semantic annotations for each of the captions, by covering 300 noun and 97 verb classes. The latter provides a lower amount of training clips (10337) but still offers a challenging evaluation set with 3492 clips. While semantic annotations are not available for YouCook2 they can be computed using WordNet and the Lesk algorithm, as described in Sec. 6.3.1. Furthermore, as both EPIC-Kitchens-100 and YouCook2 share the kitchens domain, the class knowledge of the former can also be used for the latter [298].

Implementation details. For EPIC-Kitchens-100 we use the TBN [134] features from the dataset provider comprising of 25 uniformly sampled RGB, flow, and audio feature vectors for each clip. For YouCook2 we use ImageNet-pretrained ResNet-152

Method	rel-Δ	nDCG	mAP
MME		48.5	38.5
MME	✓	49.6 **↑1.1**	39.2 **↑0.7**
JPoSE		<u>53.5</u>	44.0
JPoSE	✓	**56.2** **↑2.7**	**45.8** **↑1.8**
HGR		32.2	36.0
HGR	✓	50.2 **↑18**	<u>45.6</u> **↑9.6**

Table 6.1: nDCG and mAP results on EPIC-Kitchens-100 with three different methods, using TBN (RGB, Flow, Audio) features. We report in bold the best results (and underline the second best). With "↑X" we represent an improvement of X when compared to the above result.

features from the VALUE benchmark [158]. For the three methods we use the open source codebases provided in the respective papers and follow their hyper-parameter setting. We release our code and models on GitHub to support reproducibility.

6.4.2 Relevance-based margin results

EPIC-Kitchens-100. To validate the effectiveness of the proposed relevance-based margin, we explore three methods (MME, JPoSE, and HGR as described in Sec. 6.3.3) on EPIC-Kitchens-100. In Tab. 6.1 we report nDCG and mAP values, averaged between text-to-video and video-to-text. In all three cases, we observe a large improvement in both metrics, showing that the relevance-based margin works on very different models. It also works well with both offline mining with randomly sampled triplets (for MME and JPoSE), and online mining with hard negatives (for HGR): by using the relevance-based margin, MME gains +1.1 nDCG and +0.7 mAP, JPoSE +2.7 nDCG and +1.8 mAP, and finally HGR obtains +18 nDCG and +9.6 mAP. Such a large improvement is possibly due to how the triplets are sampled: in JPoSE, the negatives do not share the verb class of the anchor, leading to a relevance lower than 0.5; but, there is not such a guarantee in HGR, since batches are formed randomly. Hence, by employing a relevance-based margin in HGR we automatically deal with situations in which the negatives have a considerable relevance and adapt the margin accordingly. Finally, in App. 6.5 we report the public leaderboard for the retrieval challenge, confirming the improvement we observe over current state-of-the-art methods.

YouCook2. In the previous experiment we used the class knowledge which accompanies the dataset. But, by computing synsets knowledge in a similar way to what is done in EPIC-Kitchens-100, the proposed relevance-based margin can still successfully help the training process. This setting poses two additional challenges: first of all, in EPIC-Kitchens-100 most of the captions follow a precise structure, i.e. they contain a verb and a noun, which is not the case when dealing with other datasets, where free-form descriptions are often adopted. This may make it more difficult for the PoS-tagger to correctly tag the words. Secondly, there may be words which are put in the wrong category by WordNet.

For this dataset, we use the same class knowledge used in EPIC-Kitchens-100, as it transfers well across both datasets since they share the cooking domain [298], and for

Method	rel-Δ	nDCG	mAP
MME		46.9	19.3
	✓	47.3 ↑0.4	19.5 ↑0.2
JPoSE		<u>49.6</u>	20.5
	✓	**50.4** ↑0.8	21.5 ↑1.0
HGR		41.0	<u>23.0</u>
	✓	46.5 ↑5.5	**26.1** ↑3.1

Table 6.2: nDCG and mAP using MME, JPoSE, and HGR on YouCook2. We use ResNet-152 (pretrained on ImageNet) features from the VALUE benchmark [158].

words which do not appear in any class, a new singleton class is created.

In Tab. 6.2 we report the nDCG and mAP values obtained with MME, JPoSE, and HGR. From the table, one can see that even in this different setting the relevance-based margin is able to provide useful information to the model. For example, the addition of the proposed technique in HGR leads to a gain of +5.5 nDCG and +3.1 mAP when compared to the results obtained with a fixed margin.

6.4.3 Ablation studies

We perform the ablation studies on EPIC-Kitchens-100 using JPoSE.

Varying the fixed margin. In Sec. 6.4.2 we show that the proposed relevance-based margin leads to improved nDCG and mAP on both EPIC-Kitchens-100 and YouCook2. But, what if one would only need to carefully tune the fixed margin to obtain similar results? To answer to this question, we focus on JPoSE and vary the fixed margin Δ in $\{0.1, 0.2, \dots, 1.5\}$ (default value used in JPoSE is 1.0). We keep the rest of the hyper-parameter setting as in [299, 45] and use the officially provided TBN features. We plot in Fig. 6.2 nDCG, mAP, average R@1 for each of the tested margins. While small margins lead to worse results overall, it can be seen that increasing the margin does not improve significantly neither the nDCG nor the mAP. Moreover, the recall rates are affected only marginally as well. When compared to the performance shown by the adoption of the relevance-based margin, it can be observed that our technique manages to achieve higher nDCG and mAP values, while also keeping similar recall rates (on average, 6.3% R@1). Finally, it is worth noticing that by using the relevance-based margin we are released from the margin hyper-parameter: this is also of practical importance, because by using a fixed margin its optimal value is not known in a testing scenario, hence one would also need to perform an expensive search on the validation set in order to achieve better performance.

Losses ablation. A peculiarity of JPoSE is that it uses multiple contrastive loss terms to learn both global- and PoS-restricted joint embedding spaces. To do so, the authors employ a global loss and a PoS-level loss, both in the cross- and within-modality settings. We perform an ablation study in Tab. 6.3 to give evidence that the relevance-based margin can be helpful even when restricting the amount of loss terms used. Note that when applying the technique to the PoS-level terms (e.g. verbs) we consider the term for the opposite PoS (e.g. nouns) in Eq. 6.1 to be 1. As shown in Tab. 6.3, the adoption of the relevance-based margin leads to an improvement of +1.6 nDCG and

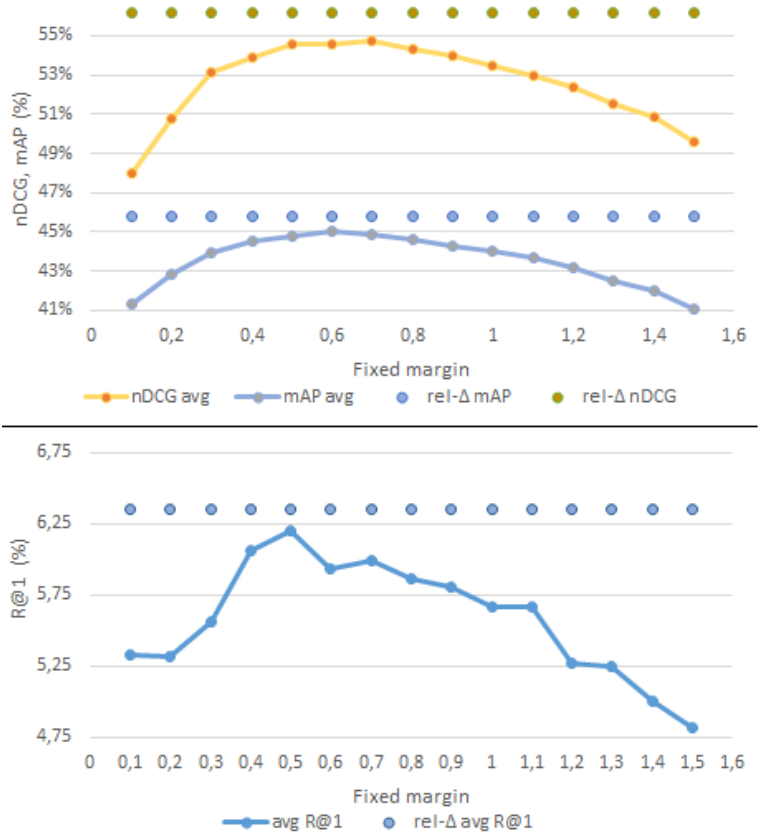


Figure 6.2: Using JPoSE on EPIC-Kitchens-100, we show how changing the fixed margin in the loss function affects the performance, measured through nDCG and mAP in the upper figure, and average R@1 in the lower one. For reference, we also plot disconnected dots to show the performance when we use the proposed relevance-based margin. Notice that the optimal fixed-margin hyper-parameter would not be known in a testing scenario; it would need to be estimated through an expensive hyper-parameter search on a validation set.

rel-Δ	cross- glob	cross- PoS	within- glob+PoS	nDCG	mAP
	✓			53.1	43.3
✓	✓			54.7 ↑1.6	44.5 ↑1.2
	✓	✓		53.4	43.7
✓	✓	✓		**56.2** ↑2.8	45.6 ↑1.9
	✓	✓	✓	53.5	44.0
✓	✓	✓	✓	**56.2** ↑2.7	**45.8** ↑1.8

Table 6.3: nDCG and mAP using JPoSE on EPIC-Kitchens-100. During training, JPoSE considers both cross- and within-modality contrastive losses, both at sentence- and PoS-level. Applying the relevance-based margin helps at each level.

Modalities	rel-Δ	nDCG	mAP
RGB		36.8	28.8
	✓	38.4 ↑1.6	30.4 ↑1.6
RGB+Flow		49.6	41.0
	✓	52.5 ↑2.9	42.8 ↑1.8
RGB+Flow+Audio		53.5	44.0
	✓	**56.2** ↑2.7	**45.8** ↑1.8

Table 6.4: TBN offers RGB, flow, and audio features. The proposed relevance-based margin interacts with each modality in an incremental way. We use JPoSE on EPIC-Kitchens-100.

+1.2 mAP when using only the cross-modal global-level loss terms, whereas +2.8 nDCG and +1.9 mAP are gained when also adding the cross-modal PoS-level terms.

Modalities ablation. For EPIC-Kitchens-100 we have RGB, flow, and audio features. To show that the improvements obtained when applying the relevance-based margin are not due to the model accessing multiple modalities related to the video, we perform another ablation in Tab. 6.4 by considering RGB-only and RGB+flow features. In both cases the proposed technique shows its usefulness. In particular, by employing the relevance-based margin we observe +1.6 nDCG and +1.6 mAP when using RGB-only, +2.9 nDCG and +1.8 mAP when using both RGB and flow, and +2.7 nDCG and +1.8 mAP when adopting all the three modalities.

6.4.4 Qualitative analysis

First of all, the proposed technique leads to variable margins, therefore the distribution of the values may help explaining why we observe such a positive influence on the final performance. In Fig. 6.3 we plot the frequencies of the margins (with bins of size 0.1) observed during the training of JPoSE on YouCook2, where for each of the training examples 10 triplets are sampled. It can be seen that a great part of the margins used are in the final bin (between 0.9 and 1.0), for which the relevance is quite low since the margin is computed as $\Delta_{a,p,n} = 1 - \mathcal{R}(a, n)$ (see Eq. 7). In these cases, the margin will be similar to the default case of JPoSE, i.e. 1.0. Yet, around 20% of the training triplets

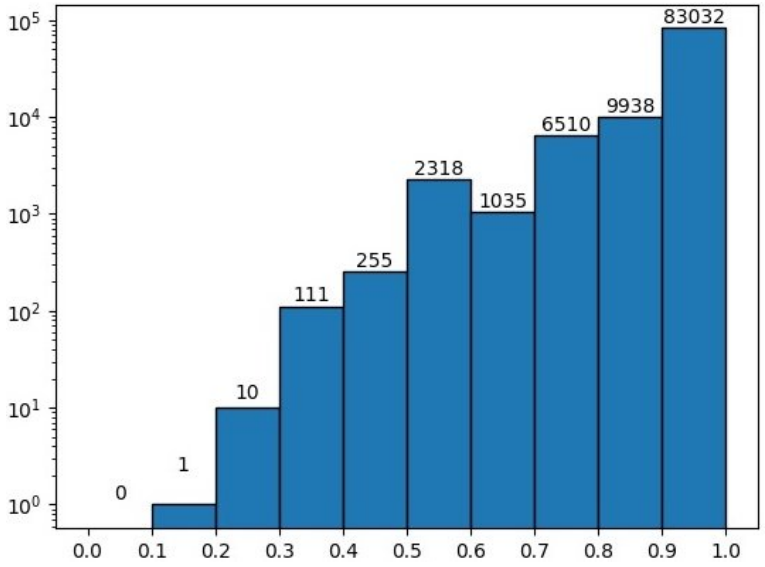


Figure 6.3: Log-scale distribution of the margins used during training. Over each bin we report the frequency. Numbers refer to one epoch, with 10 triplets sampled for each example (e.g. with 10337 examples for YouCook2, we end up with around 103k triplets). Although a great part of the triplets are separated with the highest margin (i.e. lowest relevance), around 20k triplets are distanced by various margin values.

end up having smaller margins. In these situations, the varying margins help the model achieve better performance by providing a semantic supervision on the structure of the embedding space, since the relevant items are kept at a distance which is proportional to the relevance.

Secondly, in Fig. 6.4 we visualize a few video-to-text examples from the testing set, by plotting for each of them the relevance values of each caption in both the full ranking list and the top 50 retrieved captions. By plotting the full ranking list, it is possible to see that the relevance-based margin helps improving the nDCG, as relevant captions are retrieved first. This can also be seen in the top 50 of Fig.6.4.a, 6.4.b, and 6.4.c where with the relevance-based margin no irrelevant captions are retrieved and, especially in Fig. 6.4.c, the ranking is almost ideal. Yet we can still find examples where the proposed technique fails to achieve the expected improvements. In Fig. 6.4.d, using the relevance-based margin a few irrelevant captions are retrieved, such as 'take container' and 'take milk container'. This behavior is likely related to the fact that during training captions like 'close container' and 'close milk container' are relevant (0.5) for a video depicting the action 'close fridge', since they share the same verb class. This leads to an increase in the similarity of the respective descriptors. Hence, during inference, also captions like 'take container' and 'take milk container' might have a significant similarity with the video descriptor of 'close fridge'.

Finally, we aim at further analyzing the effectiveness of the proposed technique from another point of view. To do so, we select three types of information. First of

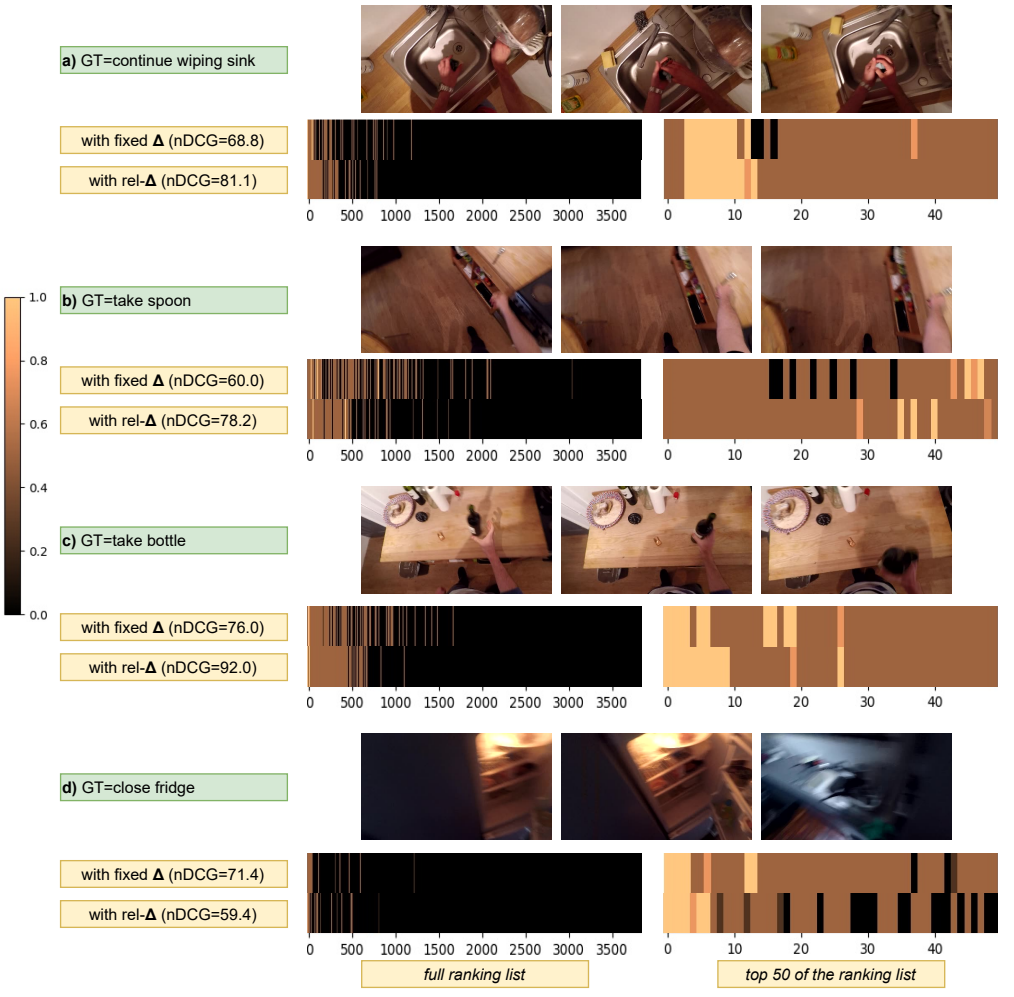


Figure 6.4: Video-to-text qualitative results on EPIC-Kitchens-100 testing set using JPoSE. For each of the examples we show a few frames and the groundtruth (GT) caption, and we plot both the full ranking list and the top 50 retrieved captions when adopting the fixed margin and then the relevance-based margin. On the left we also visualize the color bar which is used for the relevance (light colors mean high relevance, dark colors low relevance). In particular, Figures a, b, and c are success cases, whereas Figure d represents a failure case.

all, we pick a caption and compute its embedding (q), pick the corresponding video descriptor (v), and compute their similarity $s(v, q)$ through dot product. Then, we look for 10 similar captions (i.e. different captions which either share the noun or the verb class), pick the corresponding video descriptors indexed by V_+, and compute an aggregated similarity value $s(v+, q) = \frac{1}{10} \sum_{v_i \in V_+} s(v_i, q)$. Finally, we also randomly select 10 dissimilar captions (i.e. sharing neither the verb nor the first noun class), pick their video descriptors, and compute $s(v-, q)$. We compare the results using JPoSE on the testing set of EPIC-Kitchens-100, and report several examples in Figure 6.5. In Figures 6.5.a and 6.5.b the usage of a fixed margin leads to a far too high similarity of the videos in V_+ with the query q when compared to its groundtruth video descriptor v, which hurts both nDCG, mAP, and the recall rates. In Figures 6.5.c and 6.5.d the videos in V_- and those in V_+ are not properly separated, hence wrongly giving the model the impression that they are similarly relevant to the query q. In all these cases, adopting a relevance-based margin is a successful strategy to correct these wrong predictions, leading to a more robust model.

6.5 Comparison with the EPIC-Kitchens-100 Challenge leaderboard

The release of the EPIC-Kitchens-100 dataset [45] was accompanied by a public challenge for the multi-instance retrieval problem (alongside other challenges, e.g. for Action Recognition). To further prove the results we show in Section 6.4, we took part into the challenge by employing the proposed relevance-based margin on the JPoSE method [299] (see Section 3). We show the results of both the participants at the time of submission and those that took part into the previous challenge (which ended in November 2021) in Figure 6.6. The previous best result was obtained by Hao et al. (more details in the technical report [46]), which achieved on average 44.23% mAP and 53.56% nDCG. As can be seen, we achieve 45.86% mAP (+1.63%) and 56.21% nDCG (+2.65%).

6.6 Conclusions

Learning a joint embedding space using a margin-based contrastive loss is the dominant approach to deal with text-video retrieval. In the literature it is shown that by using such a framework, competitive performance on rank-unaware metrics, such as the recall rates, can be obtained. Yet, rank-aware metrics, such as the nDCG, need to be taken into account, as multiple descriptions can have numerous levels of relevance to a given query [298]. In this Chapter, we proposed to move away from the fixed margin which is typically used in such a framework, and introduced a relevance-based margin. In particular, we adopted the proposed technique into three increasingly more complex models on two datasets and gave empirical evidence that we can easily improve the performance measured through nDCG and mAP. Moreover, we showed that even by performing an expensive search of the fixed margin hyper-parameter, it does not reach the same performance as when using the relevance-based margin. Furthermore, the proposed technique can also have a positive impact on video retrieval applications as not needing to tune the margin can lead to less GPU hours required to fully train the model.

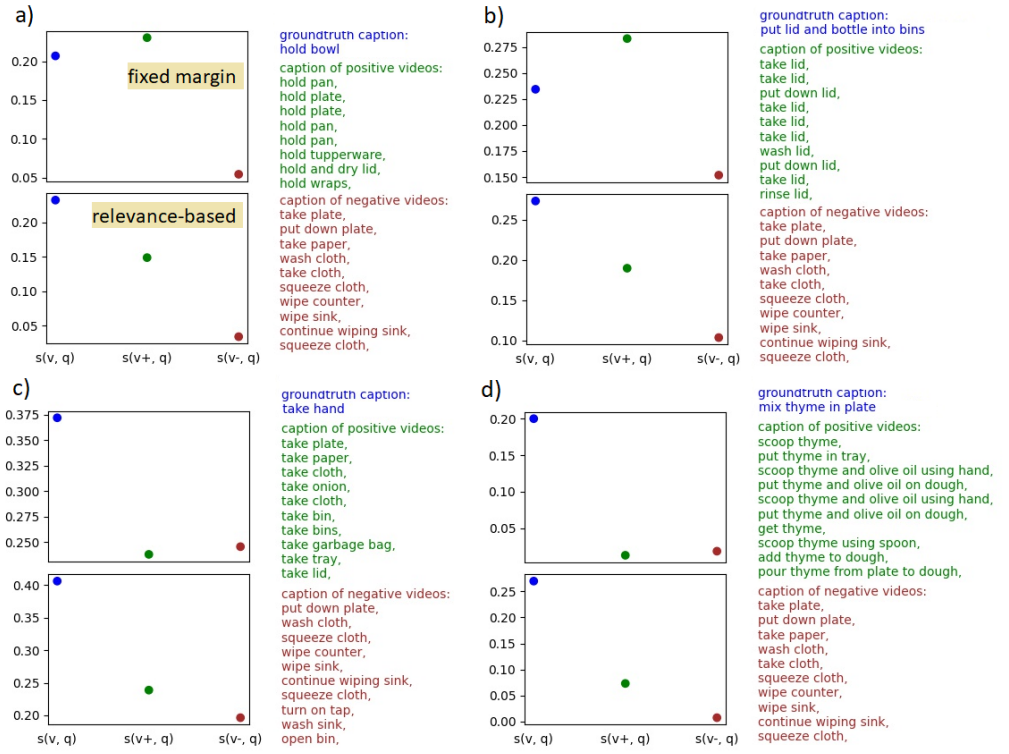


Figure 6.5: Using JPoSE, we compute a similarity score $s(v, q)$ for the groundtruth pair (colored in blue), $s(v+, q)$ for videos with similar captions (colored in green), and $s(v-, q)$ for videos with dissimilar captions (colored in brown). Note that, when selecting V_+, for the examples on the left we change the noun class, whereas on the right we change the verb class. See Sec. 6.4.4 for more details. The captions of the videos used are reported on the right. Each of the four examples are taken from EPIC-Kitchens-100 testing set and for each of them we report first what happens with fixed margin, then with the proposed relevance-based margin.

Test set 2022 Challenge (currently open)

#	User	Entries	Date of Last Entry	SLS			Mean Average Precision (%)			Normalised Discounted Cumulative Gain (%)		
				PT ▲	TL ▲	TD ▲	Avg. ▲	T2V ▲	V2T ▲	Avg. ▲	T2V ▲	V2T ▲
1	afalcon	1	01/28/22	2.00 (1)	3.00 (1)	3.00 (1)	45.86 (1)	40.36 (1)	51.36 (1)	56.21 (1)	54.23 (1)	58.19 (1)
2	MI-MM	1	12/10/21	2.00 (1)	3.00 (1)	3.00 (1)	27.58 (2)	23.08 (2)	32.09 (2)	42.10 (2)	40.48 (2)	43.72 (2)

Test set 2021 Challenge (closed)

#	User	Entries	Date of Last Entry	Team Name	SLS			mean Average Precision (%)			normalised Discounted Cumulative Gain (%)		
					PT ▲	TL ▲	TD ▲	Avg. ▲	T2V ▲	V2T ▲	Avg. ▲	T2V ▲	V2T ▲
1	haoxiaoshuai	6	04/08/21	IIE_MRG	2.0 (1)	3.0 (1)	3.0 (1)	44.23 (1)	38.49 (1)	49.96 (1)	53.56 (1)	51.83 (1)	55.28 (2)
2	JPoSE	3	01/07/21		2.0 (1)	3.0 (1)	3.0 (1)	44.01 (2)	38.11 (2)	49.91 (2)	53.53 (2)	51.55 (2)	55.51 (1)
3	MLP	6	01/06/21		2.0 (1)	3.0 (1)	3.0 (1)	38.49 (3)	33.99 (3)	42.99 (3)	48.49 (3)	46.92 (3)	50.05 (3)
4	MI-MM	4	05/06/21		2.0 (1)	3.0 (1)	3.0 (1)	29.21 (4)	23.60 (4)	34.83 (4)	44.79 (4)	42.40 (4)	47.18 (4)

Figure 6.6: We report the public leaderboard for the EPIC-Kitchens-100 Challenge at time of submission (**below**), and also the leaderboard for the previous challenge which ended in November 2021 (**above**). It can be seen that we achieve around +1.6% mAP and +2.6% nDCG over the previous best results, achieved by Hao et al. (details in the technical report [46]).

Finally, we focused our experiments on text-video retrieval, but the relevance-based margin can be easily extended to other domains where similar margin-based ranking losses are used, e.g. in image retrieval [335]. Moreover, we showed the effectiveness of the proposed approach by applying it to loss functions where the margin is explicitly defined and used to separate positive and negative pairs, e.g. [247, 29]. Yet, there are also popular loss functions which do not make use of it, such as NCE [92] and MIL-NCE [193]. Future work is required to adapt the relevance-based margin to non-margin based loss functions.

7

Learning video retrieval models with relevance-aware online mining

7.1 Introduction

When looking for a video via a multimedia search engine, the user usually describes its expected contents by means of a natural language query. Usually, the multimedia search engine responds with a ranking list of visual items, which, according to the underlying decision-making system, best fit the contents described by the given user query. Commonly, the models for text-video retrieval are created by leveraging the contents of both the video clip and the captions associated to it in a training dataset. Most of the state-of-the-art methods build a joint textual-visual embedding space via deep learning, e.g. [26, 57]. In particular, the underlying neural network organizes such a space by learning to produce a similar representation both for a video clip and the caption which describes it: by doing so, the representation of the sentence is aligned to that of the video, making them share the joint visual-textual embedding space. This makes it possible to naturally use a textual query, mapped into the same embedding space, to obtain a ranking of all the videos, and vice versa the ranking of the captions can be obtained by using a video. To learn a model capable of producing representations in a joint text-video embedding space, a peculiar type of loss functions, called *contrastive* loss functions, is often used [36, 205, 247]. These functions aim at maximizing the similarity of videos and captions which are paired in the dataset, and minimize that of examples which do not. In particular, given a video, its caption represents a *positive* example for it, whereas all the other captions form the *negative* examples.

A great effort was spent by the community on developing methodologies to select the negative examples, e.g. by selecting - or *mining* - one [247], two [29], or more negatives [262]. Other researchers focused on identifying which negatives contribute to the loss: in particular, those which contribute the most are called *hard* negatives,

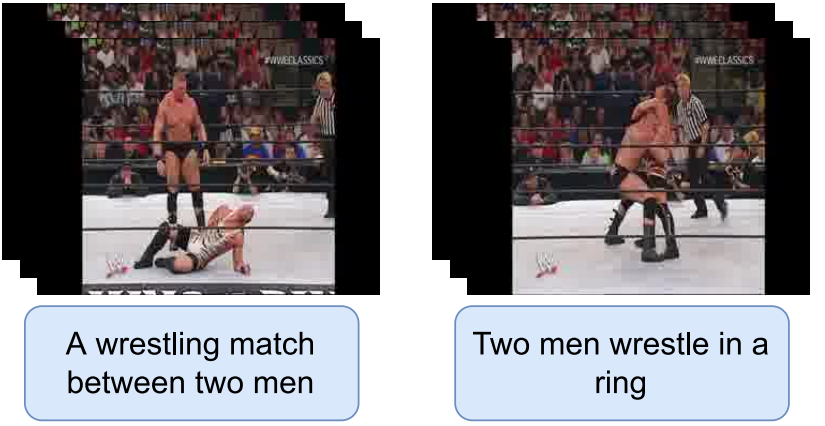


Figure 7.1: Two examples video clips from the MSR-VTT dataset [310].

whereas those which contribute highly but are not more similar to the query than a positive example form the *semi-hard* negatives [247]. Noteworthily, all these techniques assume that videos and captions which are not paired in the dataset can be used as negative examples. This assumption hardly holds in real world scenarios. Consider the example in Figure 7.1, which is commonly found in many public datasets. By following such an assumption, the second caption could be chosen as a negative for the first video, therefore forcing their descriptors to be different at training time. Yet, by looking at the videos and by reading their captions, it is clear that no real difference in their semantics is present and they should not be contrasted.

Meanwhile, the community investigated less on the usage of positive examples mainly because of how video-text datasets are usually built: in fact, there is no groundtruth label to define two videos as semantically similar, as for those in Fig. 7.1. A first attempt, involving the creation of action labels, was proposed in [298, 299] to perform the mining of both positives and negatives *offline*, that is by selecting them without taking into account their contribution to the loss.

To address these issues, in this Chapter we introduce a novel strategy to effectively improve the *online* selection of positives and negatives by leveraging the overlap of semantic concepts shared by the captions attached to the videos. Since the selection is performed by leveraging the concept of relevance, we called our strategy *RANP*, or Relevance-Aware Negatives and Positives mining. By doing so, it is possible to select the negatives avoiding the situation shown in Fig. 7.1. Moreover, we also introduce a novel strategy to identify the captions which are not related to a video in the dataset but share similar semantics with it, and use them as positive examples in a properly reformulated loss function.

The proposed strategy can be applied to two different state-of-the-art methods and to two commonly used contrastive loss functions. As regards the methods, we first validate the strategy on a method using hierarchical learning and graph reasoning [26], and then on a Transformer-based method [257]. As for the losses, we consider the Triplet loss, which maximizes the cosine similarity of a query and a positive example, while enforcing a margin to the similarity between the query and one negative at a

time [247]; and the NCE loss defined in [193] which maximizes the similarity of the positive pair while minimizing that of all the negatives within the batch. We conduct an extensive experimental analysis on two public datasets, EPIC-Kitchens-100 [45] and MSR-VTT [310]. In particular, we perform several ablation studies and achieve state-of-the-art results on both datasets, showing the robustness of our strategy. The qualitative analysis of text-to-video examples from the testing set show that models trained with our strategy generalize well and produce ranking lists which have many more highly relevant samples at the top. Moreover, we provide evidence that our technique successfully reaches state-of-the-art results by using hard, semi-hard, and all the negatives, presenting the flexibility of the proposed strategy. Finally, to support reproducibility, we publicly release the code supporting both models and loss functions at https://github.com/aranciokov/ranp.

A preliminary version of this Chapter was published as [70]. In this Chapter we extend our strategy to an additional and highly different method [257], and to an additional loss function [193], further validating its robustness and flexibility. Moreover, we achieve new state-of-the-art results on MSR-VTT, perform qualitative analyses to understand its behavior, and finally we show the flexibility of the proposed strategy by analyzing how using hard, semi-hard, and all the negatives affect the final performance.

After this introduction, in Section 7.2 we present the related works on cross-modal video retrieval and to contrastive losses. In Section 7.3 we describe a popular contrastive loss, providing details about an important shortcoming which limits their semantics awareness. Then, to address this limitation, in Section 7.4 we introduce our training methodology, whereas several experiments and analyses are discussed in Section 7.5. Finally, Section 7.6 concludes the Chapter.

7.2 Related work

Video retrieval. Cross-modal text-video retrieval is usually dealt with by learning a joint textual-visual embedding space [26, 40, 57, 257]. Since videos are composed of many modalities, many techniques to learn joint representations were introduced [76, 175, 194, 201, 288]. For instance, MoEE [194] and T2Vlad [288] are based on NetVLAD [6], whereas Collaborative Experts (CE) [175] introduced a gating mechanism for the visual and audio-related features directed by several pretrained experts. Multimodal Transformers were used in MMT [76] and Everything-at-once [257]. From the language point of view, TeachText [40] leveraged the availability of multiple language models. Differently from these, several works focus on learning structured embeddings following the structure of the input data, e.g. by working on the part-of-speech (JPoSE [299]) or by learning global and local representations via semantic roles (HGR [26]). However, the community mostly focused on instance-based video retrieval, assessing the performance by ignoring the quality of the full ranking list. Wray et al. [298] introduced a 'semantic' version of video retrieval, which uses rank-aware metrics to perform the evaluation.

Contrastive loss and mining techniques. Contrastive losses [92, 93, 100] are often used for cross-modal tasks because of their capability to maximize the descriptors' similarity for video and caption pairs in the dataset. Early works computed the loss on two samples at a time [93], whereas triplets [247], quadruplets [29], and 'N+1'-tuples [262] were later used. Yet, training on all the possible tuples from the dataset

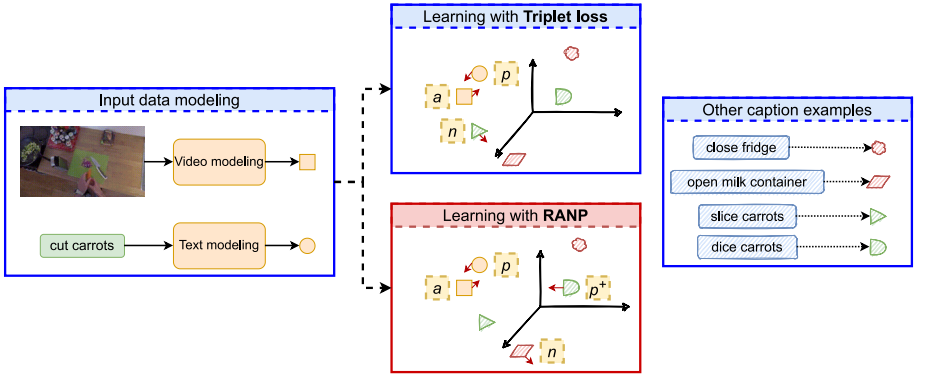


Figure 7.2: By adopting the typical approach (Eq. 7.3), a caption which is not paired to the input video is selected as hard negative based solely on its similarity. Yet, this may lead to semantically similar captions being wrongly selected as negatives, despite their high relevance to the video (see Sec. 7.3). With our proposed technique, RANP (see Sec. 7.4), we avoid this while also finding new positive captions and, consequently, the learning also increases their similarity to the video.

is unfeasible (e.g. the amount scales cubically with triplets), and many of them may not even contribute to the loss. Therefore, a subset of the tuples are selected through mining techniques, either from the dataset ('offline') or from the batch ('online'). The former is often avoided because of the need to re-sample them during training, making it burdensome. Nonetheless, it was used in several domains, e.g. deep metric learning [95, 269] and video retrieval [298, 299]. Online mining forms the tuples within the batch and is widely used [40, 57, 71, 257]. When negatives are involved in the loss computation, e.g. in [247], the loss is usually computed either on all negatives (e.g. [76, 194]), despite it may lead to some extra computation, or on a lesser amount of negatives, such as those which share a highly similar representation to the positive pair [26, 57]. Nonetheless, recent research also presented the usefulness of easy examples [313, 314]. Positive examples have also been mined in some fields, e.g. in cross-modal [100, 314] and near-duplicate video retrieval [125]), yet they use labels available in the dataset. For instance, SVD [125] contains 1206 videos in the query set for which more than 10000 video pairs are labeled as positive. An attempt which creates action labels, was proposed for offline mining in [298]. In representation learning for images or videos, positive samples were also created via transformations [28, 97, 207, 220].

Differently from previous works, we introduce semantic knowledge to the training process by computing an overlap of the semantic concepts shared among videos and captions. Moreover, we devise a two-step method to discover new positives within the batch and use them to improve the training.

7.3 Training a video retrieval model with contrastive loss and mining

Given a video $v^\star$ and a pool of candidate textual descriptions, the objective of video-to-text retrieval is to orderly retrieve each of the descriptions based on how well they describe the video, thus producing a ranking list in which such a order is given by a similarity function (computed with $s(\cdot,\cdot)$, e.g. cosine similarity). The caption paired to $v^\star$ in the dataset, $q^\star$, is expected to be ranked first. In text-video retrieval, the evaluation metrics typically focus on the rank of $q^\star$. But, the same video clip may be described by different yet semantically similar natural language captions. To capture these semantic nuances, we focus on the more recent problem of semantic video retrieval and uses rank-aware metrics, such as the Normalized Discounted Cumulative Gain [122], or nDCG, to evaluate the quality of the full ranking list. For a holistic view on the problem, the text-to-video counterpart is also considered and obtained by swapping the role of $q^\star$ and $v^\star$.

Text-video retrieval models are often trained by means of a contrastive loss, aiming at the maximization of similarity of the descriptors computed for pairs of visual and textual data. To do so, the triplet loss (Schroff et al. [247]) is often used [26, 57, 61, 76]: instead of relying solely on $s(v^\star, q^\star)$, i.e. the similarity between the anchor and the positive, it also uses a third component in the equation, called *negative*. In particular, if the video $v^\star$ is the anchor, the negative example is a textual description not paired to $v^\star$ in the dataset; conversely, if the anchor is $q^\star$, then the negative is a video clip. The triplet loss is based on this term:

$$L_n(v^\star, q^\star, q-) = max(0, \Delta_n + s(v^\star, q-) - s(v^\star, q^\star)) \tag{7.1}$$

where Δ_n is a fixed margin, and $q-$ represents the negative caption. By optimizing Eq. 7.1, Δ_n is enforced between $s(v^\star, q^\star)$ and the similarity between the video and the negative query, in order to satisfy the following constraint:

$$s(v^\star, q-) + \Delta_n < s(v^\star, q^\star) \tag{7.2}$$

The optimization of Eq. 7.1 can be performed in several ways. Typically, all the negative captions in the mini-batch are used, e.g. in [76, 194], but this means that the loss for many easy negative captions, i.e. already satisfying Eq. 7.2, is computed although they do not contribute meaningfully to it. Therefore, the selection of a single example is often preferred. To this end, the online hard negative mining selects the most similar example to the anchor within the batch, e.g. in [26, 57]. While these examples are informative to the training process, their usage from the very start may lead the optimization process to a local minimum where the model collapses [247]. To avoid it, *semi-hard* negatives, i.e. highly similar to the anchor but less than the positive example, are often preferred and can be used also to start the training process [247, 100].

7.3.1 Online hard negative mining

Online hard negative mining consists in sampling the hardest negative within the batch, and then using it to do one parameters update step to optimize Eq. 7.1. Formally, given

$(v^\star, q^\star)$ and defining Q as the set of captions in the mini-batch, the hardest negative is the most similar example to the anchor, after the exclusion of $q^\star$:

$$q- = argmax_{q \in Q \setminus \{q^\star\}} s(v^\star, q) \tag{7.3}$$

When looking for the semi-hardest, the set $\{q \,|\, s(v^\star, q) > s(v^\star, q^\star)\}$ is also excluded. Yet, these filtered sets may still contain captions which should not be considered negative: in fact, they may contain captions which, although not paired to $v^\star$, describe in part or entirely the content of the video clip. For instance, let $q^\star$ be 'cut carrots', q_1 'slice carrots', q_2 'open milk container', and $s(v^\star, q_1) > s(v^\star, q_2)$ as in Fig. 7.2. Based on Eq. 7.3, q_1 is chosen as the hardest negative because it is not $q^\star$, hence a valid candidate, and it has the highest similarity to $v^\star$. This means that for the model such a video may be described by 'cut carrots', but not by 'slice carrots', which is considered as irrelevant as 'open milk container'. Note that with our techniques we avoid these bad selections from the start by using the semantics of the data, whereas, by solely excluding $q^\star$ and letting $s(\cdot, \cdot)$ guide the mining, these situations are likely to happen all the time during training.

7.4 Proposed method: relevance-aware online mining of positives and negatives

In order to improve the selection of the negatives and the positives, we leverage the shared semantics of videos and captions: in particular, we do so by quantifying the overlap of shared semantic concepts, which are identified in terms of nouns, verbs, and their synonyms. As an example, let: (x_1) 'pick up a flowerpot and a sunflower', (x_2) 'pick an helianthus and a flowerpot', (x_3) 'pot the lily in a flowerpot', (x_4) 'put the cake in the oven'. We expect x_2 and x_1 to be quite similar ('helianthus' and 'sunflower' are synonyms), hence x_2 is highly relevant to x_1; x_3 is slightly relevant because of 'flowerpot', but the flowers and actions are different; and x_4 is irrelevant. Hence we look for a notion of relevance which captures semantic relations, such as synonyms, and use them to determine a continuous value representing how semantically close two captions are. For such a task, we adopt the definition for a relevance function $\mathcal{R}(x_i, x_j)$ given by Damen et al. [45]:

$$\mathcal{R}(x_i, x_j) = \frac{1}{2}\left(\frac{|x_i^V \cap x_j^V|}{|x_i^V \cup x_j^V|} + \frac{|x_i^N \cap x_j^N|}{|x_i^N \cup x_j^N|}\right) \tag{7.4}$$

where x_i^V and x_i^N represent, respectively, the set of verb and noun classes identified in the i-th caption. Classes are used here to include both a token, e.g. 'helianthus', and its synonyms, e.g. 'sunflower'. When x_i or x_j is a video, two situations arise. If v_i is paired to only one caption q_i in the dataset, the noun and verb classes of q_i are used for v_i. Conversely, if there are multiple captions, then a word set made of the classes which are shared among many different captions of v_i is built, as in [298]. Formally: $x_i^N = \{c^N | c^N \in \mathcal{K}(x_i)_{|\rho,N}\}$, where c^N is a noun class, $\mathcal{D}(x_i)$ represents the captions of x_i, and $\mathcal{K}(x_i)_{|\rho,N}$ is the set of classes for the part-of-speech N appearing in at least $\rho \cdot |\mathcal{D}(x_i)|$ captions. Formally, $\mathcal{K}(x_i)_{|\rho,N}$ is defined as $\{c \,|\, PoS(c) = N \wedge |\{d | d \in \mathcal{D}(x_i) \wedge c \in d\}| \geq \rho \cdot |\mathcal{D}(x_i)|\}$, where $PoS(\cdot)$ predicts the part-of-speech of a class. The

same steps are used for x_i^V. Finally, looking at the example, the following are computed: $\mathcal{R}(x_1, x_2) = 1$, $\mathcal{R}(x_1, x_3) = 0.16$, and $\mathcal{R}(x_1, x_4) = 0$.

7.4.1 RAN: Relevance-aware online hard negative mining

In Sec. 7.3.1 we provide an intuitive description of a shortcoming of current techniques used to mine negative examples. Formally, if we define a threshold τ to separate irrelevant from relevant content, then $\{q \,|\, \mathcal{R}(v^\star, q) \geq \tau,\, q \in Q \setminus \{q^\star\}\}$ may be non empty: as a consequence, a relevant caption may be selected as a negative, leading to the aforementioned shortcoming. Note that the same holds for semi-hard negative mining. Then, by optimizing the triplet loss, the similarity of $q-$ to $v^\star$ gets decreased, although $q-$ describes at least part of the contents of $v^\star$, given its positive relevance. We address this issue by introducing RAN, which binds the sampling procedure to the relevance function, hence avoiding the selection of a 'false negative'. This is done by improving Eq. 7.3 and by filtering the relevant content from the negatives' pool, by the following equation:

$$q- = argmax_{q \in Q \setminus \{q \,|\, \mathcal{R}(v^\star, q) \geq \tau\}} s(v^\star, q) \tag{7.5}$$

When using semi-hard negatives, the following set is excluded from the selection in Eq. 7.5: $\{q \,|\, \mathcal{R}(v^\star, q) \geq \tau \wedge s(v^\star, q) > s(v^\star, q^\star)\}$. Therefore, the relevance function $\mathcal{R}$ is fundamental in the sampling procedure, and the exclusion of an example is no longer based solely on its relation to $v^\star$ in the dataset.

7.4.2 RANP: Relevance-aware online hard positive mining

By using RAN, only irrelevant samples are used as negatives, and relevant ones are not seen as negatives anymore. Yet, this means that relevant captions and videos could still play a role in the optimization process, but they are not currently used. Therefore, we propose RANP, a two-steps strategy to discover additional relevant samples, thus adding positive mining which is not pursued for text-video retrieval. To do so, the first step consists in finding a relevant caption $q+$ for $v^\star$, *i.e.* $\mathcal{R}(v^\star, q+) \geq \tau$, which has a far too dissimilar representation when compared to $v^\star$. This could be given by $argmin_{q \in Q \setminus \{q^\star\}} s(v^\star, q)$ yet, by doing so, easy negative captions which are not violating Eq. 7.2 could be chosen. Therefore, the relevance function is selected as a means to identify a 'pool of positives' from which relevant samples can be mined. This pool is defined by excluding the irrelevant samples found in $\{q \,|\, \mathcal{R}(v^\star, q) < \tau\}$:

$$q+ = argmin_{q \in Q \setminus \{q \,|\, \mathcal{R}(v^\star, q) < \tau\}} s(v^\star, q) \tag{7.6}$$

As the second step, the loss is extended by including a new term which aims at increasing the similarity of $v^\star$ and $q+$. This can be done by having a new triplet loss term, where $q+$ plays the role of $q^\star$. Formally:

$$L_p(v^\star, q+, q-) = max(0, \Delta_p + s(v^\star, q-) - s(v^\star, q+)) \tag{7.7}$$

Finally, given a mini-batch of B clip and caption pairs, the final loss is computed by summing the video-to-text $\mathcal{L}_{v-t}$ and the text-to-video loss $\mathcal{L}_{t-v}$. In particular, $\mathcal{L}_{v-t}$ is

defined as:

$$\mathcal{L}_{v-t} = \frac{1}{|B|}\Big(\sum_{\substack{v \in B \\ q+ \leftarrow Eq.7.6 \\ q- \leftarrow Eq.7.5}} L_p(v^\star, q+, q-) + \sum_{\substack{v \in B \\ q- \leftarrow Eq.7.5}} L_n(v^\star, q^\star, q-)\Big) \tag{7.8}$$

whereas $\mathcal{L}_{t-v}$ is computed by switching v and q.

7.5 Experimental results

We validate our methodology on two large scale vision and language datasets: EPIC-Kitchens-100 [45] and MSR-VTT [310]. **EPIC-Kitchens-100** [45] provides 67217 clips for training and 9668 for testing. A set of 4834 clips from the training set is used for validation and fast assessment of the performance, as done in the retrieval baselines of [45]. Each clip is annotated by a brief caption describing an activity in the kitchen, and by verb and noun semantic classes. **MSR-VTT** [310] comprises 10000 clips about multiple scenarios, each annotated by 20 free-form captions. We follow the official split (from [310]) of 6513, 497, and 2990 clips for training, validation, and testing. For MSR-VTT, we compute the semantic classes with a pipeline made of spaCy, WordNet [198], and Lesk algorithm [155] as in [298]. We use $\rho = 0.25$ (see Sec. 7.4).

We consider two recent methods for all the experiments. **HGR** [26] performs graph reasoning on hierarchical representations of video and caption. To cover recent state-of-the-art Transformer-based methods for text-video retrieval, we also include **Everything-at-once** in its text-video version [257]. On both datasets, the training lasts for 50 epochs with a batch size of 64. For EPIC-Kitchens-100 we use officially provided TBN features [45], whereas for MSR-VTT we use ImageNet-pretrained ResNet-152 features and Kinetics400-pretrained 3D-ResNeXt-101 features from [257].

The evaluation on the testing set is performed with the best validation model. We use two metrics for evaluation, Normalized Discounted Cumulative Gain (nDCG) [122] and Mean Average Precision (mAP) [10], as in [298]. For MSR-VTT we do not use mAP because, due to the different computation of semantic classes for the videos, the relevance values are always lower than one. More details in the Supplementary.

7.5.1 Distribution of relevance among the hard negatives

Fig. 7.3 presents the distribution of relevance values among the hard negatives selected in one epoch of training. On EPIC-Kitchens-100 (Fig. 7.3.a) more than 50% of the negatives have a positive relevance to the input caption, and 13% of them are 100% relevant, that is they share the same noun and verb classes. Overall, four modes for the relevance values are identified: 0 (45%), 50 (36%), 100 (13%), 25 (3%). By using RAN, the distribution changes considerably: for instance, by using $\tau = 0.75$ (orange bar) it changes as in Fig. 7.3.b, avoiding negatives which are highly relevant (>75%).

As mentioned in Sec. 7.4, for MSR-VTT the video classes are chosen among those

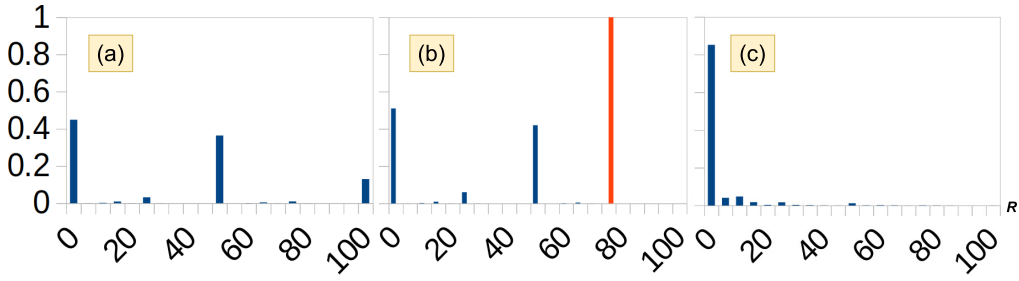


Figure 7.3: Distribution of relevance values among the hard negatives mined in one epoch (batch size 64) on EPIC-Kitchens-100 (**a**) and MSR-VTT (**c**). In (**b**), we visualize the change when using RAN with $\tau = 0.75$ (orange bar).

which appear in at least ρ, e.g. 25%, of the captions paired to that clip. It follows that even the captions paired to a video may not be 100% relevant: thus, finding relevant samples within random mini-batches is more difficult (Fig. 7.3.c), leading to much lower modes: 0 (85.2%), 10 (4.7%), 5 (4%), 15 (1.7%).

7.5.2 Influence of the threshold τ on the proposed techniques

The previous analysis is useful to highlight some peculiar values which can be used for τ. Therefore we use them, along HGR and the proposed training strategies, reporting the results in Fig. 7.4: for EPIC-Kitchens-100 we visualize (7.4.a) the nDCG and (7.4.b) the mAP, whereas for MSR-VTT (7.4.c) the nDCG. On both datasets we keep $\Delta_n = \Delta_p = 0.2$ (as in [26]), since by changing Δ_n and Δ_p only small changes in performance are achieved (experiments in Supplementary). Noteworthily, compared to the original model, the usage of the proposed strategies consistently leads to improvements.

According to the previous analysis, lower values of τ should avoid the selection of several 'bad negatives' and help the training process. This is confirmed by Fig. 7.4, since the lower the value for τ, the higher the performance on both metrics. For instance, $\tau = 0.15$ avoids more than 50% of bad selections on EPIC-Kitchens-100 and, in fact, around +23% nDCG and +8% mAP is measured. Detailed results in Supplementary.

Secondly, by training HGR with RAN, it achieves up to +12.9% nDCG (48.8%) and +7.0% mAP (46.5%) over the original model; on MSR-VTT it improves by up to +3.4% nDCG (28.7%). The improvements on EPIC-Kitchens-100 are far higher thanks to the simplicity of the captions, which allows for an easier search and removal of relevant captions from the negatives' pool. Conversely, the relevance values for the captions in MSR-VTT are generally smaller, due to how they are computed (see Sec. 7.4), therefore it is more difficult to find relevant captions.

Finally, the additional positive example found with RANP leads to further improvements on both datasets, since it helps pushing to the top of the ranking lists multiple captions and videos which are relevant to the query. On EPIC-Kitchens-100, they measure up to +23.1% nDCG (with $\tau = 0.40$) and +7.7% mAP (with $\tau = 0.15$), whereas up to +5.8% nDCG is observed on MSR-VTT with $\tau = 0.10$. Based on these observations,

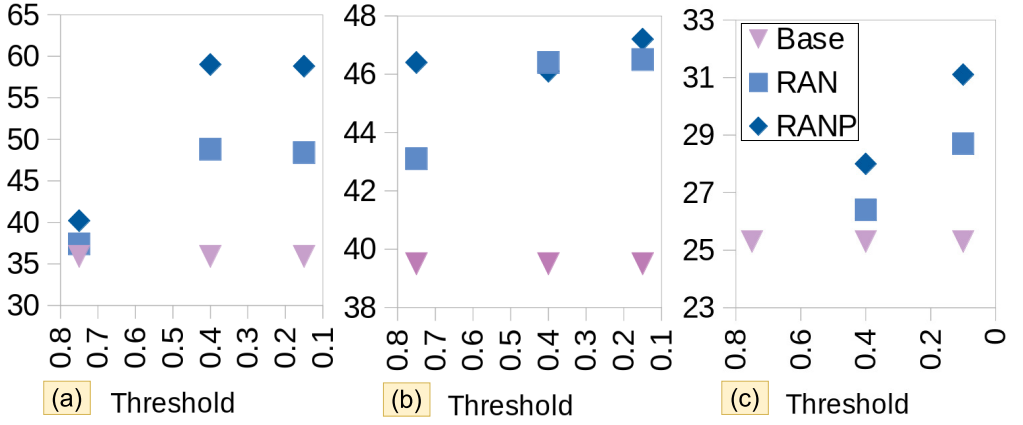


Figure 7.4: We compare the influence of RAN and RANP on the performance obtained by HGR. On EPIC-Kitchens-100: (a) nDCG (b) mAP; on MSR-VTT (c) nDCG. Discussion in Sec. 7.5.2.

in the following experiments we use $\tau = 0.15$ for EPIC-Kitchens-100 and $\tau = 0.10$ for MSR-VTT.

7.5.3 About the optimization strategy for negatives

RANP serves as a strategy to separate *relevant* negatives from those *irrelevant*. Then, the optimization is often performed by looking at the negatives which produce the highest loss: intuitively, these hard examples should be the most useful to guide the training, yet as mentioned in Schroff et al. starting the training with them may also lead to a bad local minimum [247]. To this end, they proposed to use semi-hard negatives, but using all the negatives in the batch could also be a solution, although it leads to heavier and potentially useless computations because many negatives may produce a null loss. To investigate the differences in performance achieved by these different strategies and how they interact with the additional constraints imposed by RANP, we explore four of them: *All* refers to the usage of all the negatives; *Semi+All* means that initially semi-hard negatives are used, followed by an additional training with the All strategy; *Semi+Hard* starts with semi-hard negatives, and then uses hard negatives; finally, *Hard* means that hard negatives are used from the very start. Note that in our experiments the *Hard* strategy is only used with HGR, as it leads Everything-at-once to a collapsed model.

The results are presented in Table 7.1. Two major observations can be made. First of all, in all the considered cases the improvement obtained by using RANP is consistent and considerable, across both metrics, models, and datasets. Secondly, on MSR-VTT the best results on both models are achieved when using the *Semi+All* strategy. Conversely, on EPIC-Kitchens-100 hard negatives are preferred, leading to 58.8% nDCG and 47.2% mAP by using HGR, and to 59.5% nDCG and 45.1% mAP by using Everything-at-once after a warmup with semi-hard negatives. Such a different behaviour may be a

Table 7.1: Influence of the negative selection strategy in the triplet loss (discussed in Sec. 7.5.3). The symbol ✓is used to mark the usage of RANP during training.

Opt	RANP	MSR-VTT	EPIC-Kitchens-100	
Everything-at-once [257]		nDCG (%)	nDCG (%)	mAP (%)
All		24.8	34.5	35.0
All	✓	**33.6**	**57.7**	**41.6**
Semi+All		21.3	33.3	33.9
Semi+All	✓	**34.4**	**58.6**	**46.0**
Semi+Hard		20.6	32.7	33.5
Semi+Hard	✓	**29.9**	**59.5**	**45.1**
HGR [26]		nDCG (%)	nDCG (%)	mAP (%)
All		26.7	37.1	40.8
All	✓	**34.4**	**57.5**	**42.4**
Semi+All		26.0	34.9	39.1
Semi+All	✓	**35.4**	**55.6**	**42.8**
Semi+Hard		23.8	34.4	38.1
Semi+Hard	✓	**27.8**	**54.0**	**45.5**
Hard		25.3	35.9	39.5
Hard	✓	**31.1**	**58.8**	**47.2**

consequence of the distribution of the relevance values highlighted in Sec. 7.5.1, making harder negatives less informative in MSR-VTT. Moreover, considering that, by using the official split of MSR-VTT, the similarity of a clip with each of its captions is increased in the process, performing the optimization on all the negatives may pose an easier problem resulting in a model which generalizes better.

7.5.4 Ablation study

In Table 7.2 we present two ablation studies on MSR-VTT using HGR (Table 7.2.a) and Everything-at-once (Table 7.2.b), serving respectively as a single-modal (appearance-only) and multi-modal (appearance and motion) model. For the former, we used the *Hard* strategy, whereas *All* was used for the latter. The results provide empirical evidence that improving the selection of the negatives by using RAN already leads to an improvement over original models (+3.4% in HGR, +2.7% in Everything-at-once). By also mining positives with RANP, further improvements are achieved, leading to +6.3% over HGR and +8.8% over Everything-at-once. Note that similar results are obtained while using other strategies, e.g. +7.7% is achieved by HGR with RANP using the *All* strategy (see Table 7.1).

7.5.5 Effects of large scale pretrain

Large scale pretraining is often performed to leverage transfer learning and, possibly, to ease the training process. In Shvetsova et al. [257] the authors pretrained their proposed model on HowTo100M [195], a large scale dataset of tutorial clips. By leveraging the pretrained weights for Everything-at-once shared by the authors, we explore the effects

Table 7.2: Ablation study on MSR-VTT using a **(a)** single-modal model (appearance-only) and a **(b)** multi-modal (appearance and motion) model. Discussion in Sec. 7.5.4.

(a) Appearance-only	nDCG		
Model	t2v	v2t	avg
HGR	24.6	26.1	25.3
HGR+RAN	27.4	30.1	28.7
HGR+RANP	**29.1**	**34.1**	**31.6**
(b) Multi-modal	nDCG		
EAO	23.9	25.6	24.8
EAO+RAN	26.4	28.6	27.5
EAO+RANP	**31.5**	**35.7**	**33.6**

Table 7.3: Influence of HowTo100M-pretrain on Everything-at-once [257] and subsequent finetuning with several strategies. Experiments performed on the official split of MSR-VTT.

PT	Opt	RANP	nDCG (%)	Mean R@1 (%)
	All		24.8	6.9
✓	All		23.1	8.6
	All	✓	33.6	4.3
✓	All	✓	34.4	5.5
	Semi+All		21.3	8.2
✓	Semi+All		21.5	8.4
	Semi+All	✓	34.4	6.7
✓	Semi+All	✓	**35.6**	6.2
PT	Details		nDCG (%)	Mean R@1 (%)
✓	*Zero-shot*		21.5	9.2
	Train w/ NCE		26.1	9.3
✓	*Fine-tune w/ NCE*		28.3	**10.5**

of such a technique in the semantic video retrieval task, both on EPIC-Kitchens-100 and on the official split of MSR-VTT.

According to the results shown in Table 7.3, starting the training process from the pretrained weights has a positive effect on both nDCG and recall rates in most of the cases. For instance, while using the NCE loss originally used by the model it leads to +2.2% nDCG (28.3%), +1.2% (9.2%) and +1.3% (+11.8%) R@1. In particular, a state-of-the-art nDCG of 35.6% is obtained by performing the finetuning with the *Semi+All* strategy and RANP.

7.5.6 Extension of RANP to NCE

In this section, we show that RANP may be extended to other contrastive loss functions, making them more suitable for the semantic text-video retrieval task. We focus on the softmax version of the NCE as detailed in Miech et al. [193] and also used in Shvetsova

Table 7.4: Experiments using the extension of NCE to a RANP-like strategy (called NCE-RANP here). See Sec. 7.5.6 for details. For MSR-VTT, the influence of the pretrain is also tested, confirming previous observations (see Sec 7.5.5).

	MSR-VTT		EPIC-Kitchens-100	
NCE-RANP	w/o PT nDCG	w/ PT nDCG	nDCG	mAP
	26.1	28.3	36.2	37.4
✓	**28.7**	**31.1**	**57.1**	**41.4**

et al. [257]. It is defined as follows, with temperature ρ and batch size B:

$$\mathcal{L}(v_j, q_j) = -log\frac{\exp(v_j^T q_j/\rho)}{\sum_{i=1}^{B}\exp(v_j^T q_i/\rho)} \tag{7.9}$$

This is also done in the opposite direction, hence normalizing $q_j^T v_j$ with respect to all the v_i. RANP aims at identifying the hardest positive (Eq. 7.6) and increasing its similarity to the video (Eq. 7.7). To bring this idea into Eq. 7.9, an additional term which uses the newly found positive in place of the original q_j, may be introduced as follows:

$$\begin{aligned}\mathcal{L}_{NCE-RANP}(v_j, q_j) = &- log\frac{\exp(v_j^T q_j/\rho)}{\sum_{i=1}^{B}\exp(v_j^T q_i/\rho)} \\ &- log\frac{\exp(v_j^T \text{q+}/\rho)}{\sum_{i=1}^{B}\exp(v_j^T q_i/\rho)}\end{aligned} \tag{7.10}$$

where q+ is selected as in Eq. 7.6. Table 7.4 shows the results obtained by using this variation, which we call NCE-RANP. On the official split of MSR-VTT, its usage leads to an improvement of more than 2.6% nDCG, reaching up to 31.1% nDCG with the pretraining. Secondly, on EPIC-Kitchens-100 we also observe a considerable improvement, reaching 57.1% nDCG (+20.9%) and 41.4% mAP (+4%).

7.5.7 Qualitative analysis

A qualitative analysis is performed on the testing set to understand and visualize how the training with RANP affects the final performance. Fig. 7.5 shows the full ranking list of the 9668 clips produced by the four models (HGR and Everything-at-once, trained with or without RANP) for three different queries on EPIC-Kitchens-100. By training with RANP, the ranking lists produced on the test set have most of the relevant videos at the top of the ranking list, e.g. it can be clearly seen in the first two queries, "wipe counter" and "put down bins". In the third query, "put tablecloth into cupboard", it can still be observed that more relevant videos have low ranks than in the two models trained without RANP. Nonetheless, some highly relevant videos have high ranks, e.g. in HGR trained with RANP there is one near the middle of the list.

Fig. 7.6 presents a more detailed visualization of the top 5 retrieved videos. As in

Table 7.5: Comparison on state-of-the-art methods for EPIC-Kitchens-100. In-depth discussion at Sec. 7.5.8. Ego-VLP and UniUD-UB-UniBZ are not fairly comparable to our method, since the former uses a huge amount of additional egocentric data (3 millions annotated clips), and the latter uses an ensemble of models, including HGR+RANP.

EPIC-Kitchens-100	nDCG (%)			mAP (%)		
Model	t2v	v2t	avg	t2v	v2t	avg
HGR [26]	37.9	41.2	35.9	35.7	36.1	39.5
EAO [257]	35.2	37.3	36.2	33.9	40.8	37.4
MME [299]	46.9	50.0	48.5	34.0	43.0	38.5
JPoSE [299]	51.5	55.5	53.5	38.1	49.9	44.0
Hao et al. [46]	51.8	55.3	53.5	38.5	50.0	44.2
IIE-MRG [244]	54.1	56.6	55.3	38.1	47.5	42.8
Falcon et al. [71]	-	-	56.2	-	-	45.8
Ours						
EAO+RAN	37.9	38.6	38.2	35.5	41.2	38.3
HGR+RAN	47.1	49.7	48.4	43.1	49.9	46.5
EAO+RANP	**57.5**	**61.6**	**59.5**	39.6	50.6	45.1
HGR+RANP	56.5	61.2	58.8	**42.3**	**52.0**	**47.2**
Ego-VLP [169]	59.6	63.3	61.4	41.0	53.9	47.4
UniUD-UB-UniBZ [69]	58.9	63.2	61.0	44.4	55.2	49.8

the previous case, we explore three queries on EPIC-Kitchens-100. By looking at the query "continue wiping sink", Fig. 7.6.a shows that HGR trained both with or without RANP, and Everything-at-once trained with RANP are able to retrieve highly relevant examples. In particular, the fifth video retrieved by "HGR base" (which is also the third retrieved by "EAO+RANP") has similar appearance features but a relevance of 0.50 because it is captioned by "wipe off kitchen", making its noun class unrelated to the more precise "sink". Conversely, without RANP, Everything-at-once retrieves some videos which have similar motion but less precise appearance features (second, fourth, and fifth are not about "sinks"). In Figures 7.6.b and 7.6.c the lists are more varied, although displaying some advantages in models trained with RANP, such as in Fig. 7.6.b where both HGR and Everything-at-once are able to retrieve more highly relevant videos in the top five. It is interesting to observe how training with RANP may lead to ranking lists in which the clips have a moderate relevance, despite not sharing the visual features: for instance, in Fig. 7.6.c "HGR+RANP" retrieves clips about "cutting" (same verb), whereas without RANP it looks for "bags" which are "opened", though not "cut" through.

7.5.8 Comparison with state-of-the-art

In Tables 7.5 and 7.6 we report the results obtained with our strategies and compare them to other methods, both on EPIC-Kitchens-100 and MSR-VTT. In particular, we report the results obtained with the *Hard* strategy for HGR and with *Semi+All* for Everything-at-once (EAO in tables).

EPIC-Kitchens-100. In Table 7.5 we compare to MME and JPoSE, proposed

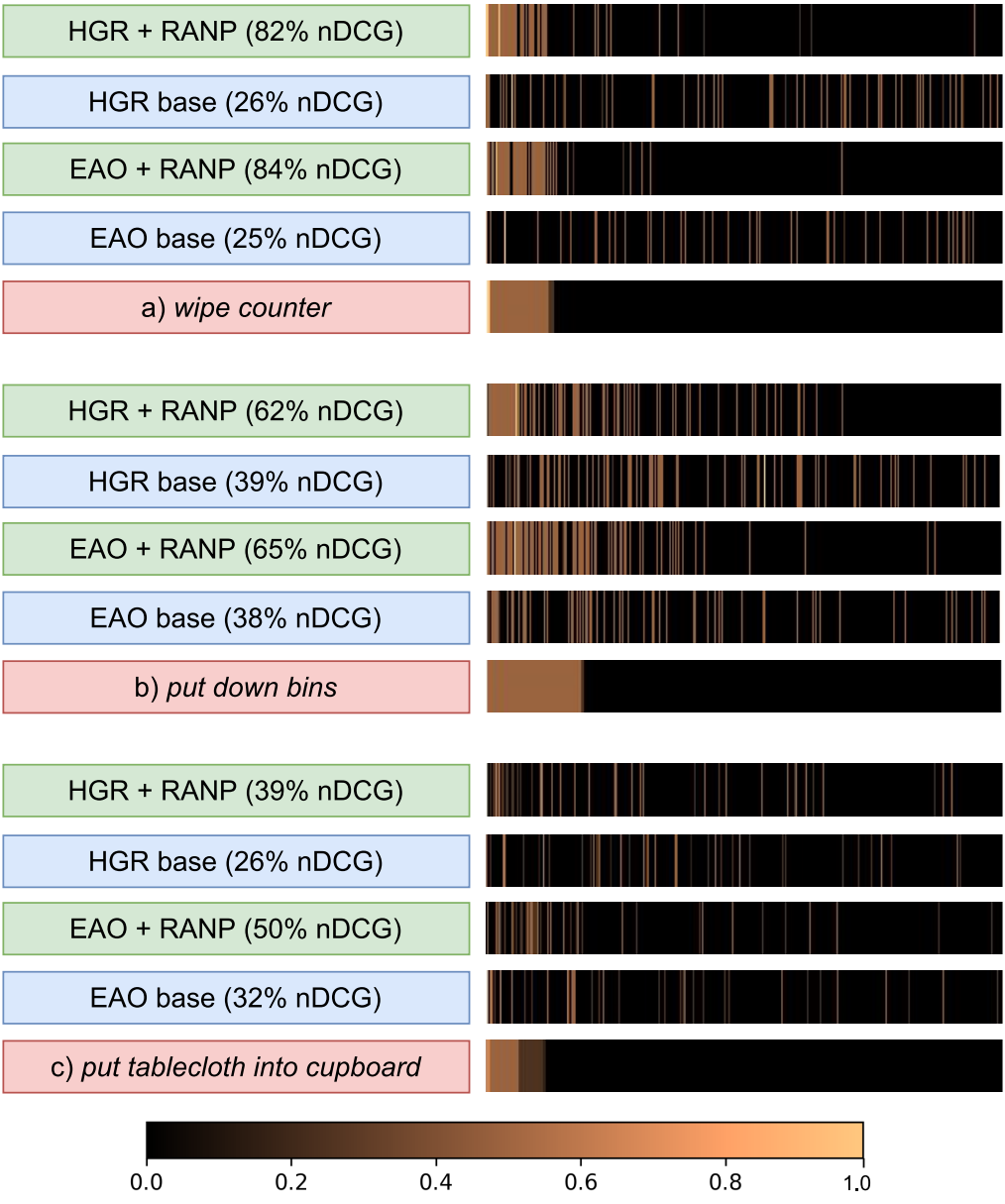


Figure 7.5: Qualitative examples of text-to-video retrieval from the testing set. The full ranking lists (of length 9668) are shown and the (color scale on the left) color represents the relevance to the query. The query is shown in red, along the optimal ranking lists. Both HGR and Everything-at-once display a similar behavior in bringing many more relevant videos to the top of the ranking list. More details in Sec. 7.5.7.

Figure 7.6: Qualitative examples of text-to-video retrieval from the testing set of EPIC-Kitchens-100. The border is colored green, yellow, orange, brown, or red based on the relevance to the query (respectively, 1.00, 0.75, 0.50, 0.25, 0.00). Discussion in Sec. 7.5.7.

by Wray et al. [299] and used in [45] as the baselines for the challenge. We include Hao et al. [46] from the 2021 edition [46], IIE-MRG [244], UniUD-UB-UniBZ [69], and Ego-VLP [169] from the 2022 edition of the EPIC-Kitchens-100 Challenge [47]. Moreover, we also include the method by Falcon et al. [71]. It needs to be noted that the two methods which currently and jointly hold the state-of-the-art, i.e. UniUD-UB-UniBZ and Ego-VLP, are not fairly comparable to our method, since the former uses an ensemble of several methods, including HGR trained with RANP, and the latter performs an additional pretraining with more than 3 millions of egocentric clips. Therefore, we compare to the current single-model state-of-the-art, that is the method proposed in [71]. Both on HGR and Everything-at-once, the addition of RANP leads to considerable improvements: HGR+RANP obtains around +2.6% (58.8% compared to 56.2%) nDCG and +1.4% mAP (47.2% compared to 45.8%); on Everything-at-once the improvement measures up to +3.3% (59.5%) nDCG, yet the previous state-of-the-art maintains a small margin of +0.7% mAP. The comparison between our two RANP-trained methods shows that Everything-at-once leads to higher nDCG (59.5% vs 58.8%), whereas HGR achieves higher mAP (47.2% vs 45.1%), meaning that the latter allows to retrieve more highly relevant captions and videos to the top of the ranking list. This may be due to the hierarchical learning aspect of HGR, which can be quite important in EPIC-Kitchens-100 considering the structure of the available captions.

MSR-VTT. For MSR-VTT, we compare to MoEE (Miech et al. [194]), CE (Liu et al. [175]), HGR, and Everything-at-once. We report for each of these models the amount of modalities used, since each differs in this regards. We evaluated CE and MoEE both using only appearance features and using all the seven available modalities within the open source codebase of [175]. Both these models, even by using one modality, achieve higher scores (29.0% and 29.4%) than HGR and Everything-at-once (25.3% and 26.1%): considering that the latter two perform better when looking at instance-based metrics, such as the recall rates, this shows that semantic video retrieval needs to be dealt with by using different tools and strategies. In fact, if HGR and Everything-at-once are trained with RANP considerable improvements are observed, respectively achieving 35.4% (+10.1%) and 34.4% (+9.3%) nDCG, which are better than the nDCG obtained by CE and MoEE trained with seven modalities (32.6% and 32.8%). Although the pretrain helps Everything-at-once achieve 35.6% nDCG, it is not as fair to be compared with the other models which are not pretrained on large scale datasets.

7.6 Conclusions

State-of-the-art methods for text-video retrieval are typically trained by using a contrastive loss, e.g. the triplet loss [247] or the NCE loss [193]. At training time, a neural network learns to output similar descriptors for each paired video and caption, while considering all the other samples as completely irrelevant. We showed this assumption hardly holds in practice, and that it leads to a suboptimal selection of negatives which share similar semantics as the query. Moreover, because only the video and caption pairs in the dataset are considered valid, there are many captions which could be used as positives for a video, but are not. To address these two shortcomings, we proposed a novel strategy, which uses the overlap of semantic concepts between captions to improve the selection of the negative examples, while also discovering new positives for a given

Table 7.6: Comparison on several state-of-the-art methods for the official split of MSR-VTT. 'Num. mod.': number of modalities used for training. 'PT': pretrain on HowTo100M.

MSR-VTT			nDCG (%)		
Model	Num. mod.	PT	t2v	v2t	avg
CE	1		28.9	30.0	29.4
MoEE	1		28.4	29.5	29.0
HGR	1		24.6	26.1	25.3
CE	7		32.2	32.9	32.6
MoEE	7		33.3	32.3	32.8
EAO	2		25.6	26.7	26.1
EAO	2	✓	27.8	28.8	28.3
Ours					
HGR+RAN	1		27.4	30.1	28.7
EAO+RAN	2		26.4	28.6	27.5
HGR+RANP	1		29.1	34.1	31.6
HGR+RANP	1		**33.0**	**37.8**	**35.4**
EAO+RANP	2		32.5	36.3	34.4
EAO+RANP	2	✓	**33.5**	**37.8**	**35.6**

query which were not originally paired to it in the dataset. We show the effectiveness of our strategy by applying it both to a graph-based [26] and to a recent state-of-the-art Transformer-based method [257] for text-video retrieval, and to two different contrastive loss functions, achieving considerable improvements over the original models. We validate our strategy on two datasets, EPIC-Kitchens-100 and MSR-VTT, conducting an extensive experimental analysis, comprising several ablation studies and successful experiments on the usage of hard, semi-hard, and all the negatives. Finally, by performing qualitative analyses on the testing set, we present evidence that the proposed strategy leads to ranking lists which include many more highly relevant examples at the top of the list when compared to models trained without it.

7.7 Appendix

7.7.1 Evaluation metrics

We mostly use two metrics for evaluation as was in [298]: the Normalized Discounted Cumulative Gain (nDCG) [122] and the Mean Average Precision (mAP) [10]. The nDCG is computed by normalizing the DCG with the Ideal DCG (IDCG), which is computed on a optimally ordered list, i.e. it follows a descending order of relevance. Following the definition given in [45], we compute the DCG as $DCG(a, Q) = \sum_{k=1}^{N_r} \frac{\mathcal{R}(a,q_k)}{log_2(k+1)}$, where a is a caption and Q is the ranking list of the captions associated to the video clips, limited to the top N_r relevant. Then, the nDCG for a is computed as $nDCG(a, Q) = \frac{DCG(a,Q)}{IDCG(a,Q)}$. Similarly, it can be computed for a video clip v and the list of all the captions. Finally, the mean of these two values is computed for all the videos and all the captions, obtaining

Table 7.7: Influence of Δ_p and Δ_n on the final performance: Δ_p is varied in {0.10, 0.15, 0.20, 0.25, 0.30}, while Δ_n is set to 0.2.

		EPIC-Kitchens-100		MSR-VTT
Δ_n	Δ_p	nDCG (%)	mAP (%)	nDCG (%)
0.20	0.10	58.7	**47.1**	31.2
0.20	0.15	59.0	46.6	31.3
0.20	0.20	59.0	46.1	31.1
0.20	0.25	**59.2**	46.0	**31.6**
0.20	0.30	59.0	45.6	**31.6**

the final nDCG which is used to report the scores:

$$nDCG(V,Q) = \frac{1}{2}\Big(\sum_{v\in V}\frac{nDCG(v,Q)}{IDCG(v,Q)} + \sum_{q\in Q}\frac{nDCG(q,V)}{IDCG(q,V)}\Big) \tag{7.11}$$

The mAP is computed in terms of the Average Precision (AP), which in turn is based on the Precision at k, $P(k)$ [10] and on $r(k)$, which is an indicator function used to discriminate relevant, i.e. with $\mathcal{R}(a,q_k) = 1$, from irrelevant examples, i.e. with $\mathcal{R}(a,q_k) < 1$. In fact, differently from nDCG which considers continuous values for the relevance, the mAP only considers two values for it, 0 or 1. The AP for a caption a is defined as $AP(a) = \frac{\sum_{k=1}^{N} P(k)\cdot r(k)}{N_r}$, where N is the length of the ranking list, including both both irrelevant and the N_r relevant examples. A similar definition holds for a video v. Finally, the mAP is computed on all the captions and all the videos, obtaining the average mAP used in the tables:

$$mAP(V,Q) = \frac{1}{2}\Big(\sum_{v\in V} mAP(v) + \sum_{q\in Q} mAP(q)\Big) \tag{7.12}$$

7.7.2 Varying the fixed margins

The final loss described in Eq. 8 includes two loss terms which impose a fixed margin on the pairwise similarities, as described in Eqs. 1 and 5. These two margins, Δ_n and Δ_p, are set to 0.2 in this Chapter. Here we aim at briefly exploring how such a value can influence the final performance. In particular, we keep $\Delta_n = 0.2$ and vary Δ_p in a wider set of values. The results are reported in Table 7.7. As for τ, we use 0.4 for EPIC-Kitchens-100 and 0.1 for MSR-VTT, since these values lead to the best average nDCG in our experiments. As reported, varying Δ_p has only a small influence on the final performance: on EPIC-Kitchens-100, it leads to small variations (up to -0.3% if decreased, up to +0.2% if Δ_p is increased) in terms of nDCG, and up to +1% mAP; on MSR-VTT it leads to up to +0.5% nDCG.

Table 7.8: The lower the threshold τ, the less relevant examples are selected as negatives, leading to better performance.

		EPIC		MSR-VTT
	τ	nDCG (%)	mAP (%)	nDCG (%)
		35.9	39.5	25.3
RAN	0.75	37.4	43.1	25.2
RANP	0.75	40.2	46.4	-
RAN	0.40	48.8	46.4	26.4
RANP	0.40	59.0	46.1	28.0
RAN	0.15/0.10	48.4	46.5	28.7
RANP	0.15/0.10	58.8	47.2	31.1

7.7.3 Detailed performance on the influence of τ

In Section 7.5.2, the influence of the threshold τ on the final performance obtained by HGR is visualized on both EPIC-Kitchens-100 and MSR-VTT in Fig. 4. The detailed results are reported in Table 7.8.

8

Conclusions and future directions

8.1 Summary and main contributions

To solve problems which deal with multiple data modalities, such as video, audio, and text, deep learning proved to be a valuable tool in recent years. In fact, by relying on deep neural networks to automatically extract discriminative features based on a target objective, hence not having to manually craft them, several breakthroughs were possible in fields such as image recognition [98, 146], skin cancer classification [65], and self-driving cars [16]. However, to obtain these results, deep neural networks need to overcome the generalization gap, which otherwise limits the performance of the trained model on different sets of data drawn from the same distribution. Since the model needs to learn representations for the input data which are general enough and not tailored to the training data, a popular way to overcome this gap consists in adding more examples to the training set, in the hope of covering those peculiar cases for which the model was not working correctly. Although for smaller datasets manual collection and annotation is a viable solution, it does not scale well, due to high costs and the need for many trained annotators. Therefore, crowdsourcing became one of the most popular ways to collect annotated data, because it raises the opportunity to create tasks which are solved by human annotators across the globe, providing annotations in exchange for small amounts of money. For instance, crowdsourcing led to the collection of popular benchmark datasets in text-video [289, 310] and text-image understanding [145]. However, recent state-of-the-art approaches often involve enormous neural networks, comprising billions of parameters, which require even bigger datasets, leading to enormous and possibly unsustainable costs for the collectors. For instance, the foundational model Flamingo [3] comprises 80 billion parameters and was trained with a mixture of text-video and text-image pairs, for a total of around 500 million pairs. To collect these impressive amounts of data, there is a need for more scalable and cheaper approaches than crowdsourcing. In particular, given the amount of multimodal data

publicly available on the Internet, e.g. the 500 hours of content uploaded to YouTube every minute [25], very large scale datasets were recently collected by crawling through the Internet and automatically downloading multimodal contents and the associated textual descriptions. Although improvements in several tasks were recently achieved by using web scraped datasets [257, 3, 19], a question naturally arises: is the collection of more and more data the only way to overcome the challenge offered by the generalization gap? On the one hand, there are many fields in which collecting more data is difficult, and annotating them is even more prohibitive due to really high costs, for instance in medical fields which require expert knowledge to provide curated labels. On the other hand, processing all these data requires an ever increasing computational power, leading to increased carbon emissions [268].

In this thesis we showed that there are indeed other ways to reduce the generalization gap and improve deep learning-based solutions without having to resolve to the collection of more annotated data. In particular, we showed that it is possible to achieve such an objective by leveraging semantic aspects of the available data which often go unnoticed.

The first part presents different approaches to create new labels which are greatly helpful at improving the generalization on different vision and language tasks. In particular, Chapter 2 introduces, for the video question answering task, three data augmentation techniques which create new questions and answer sets based on the semantics of the available annotation. These techniques were based on a statistical analysis of the annotations, which showed that some biases were affecting them, and that some positional labels (e.g. objects which are visible on the "left" of something) were too sparsely used. The proposed techniques made it possible to alleviate the highlighted biases and to create new instances of the positional labels which increased the robustness of the trained model. In Chapter 4, we pursued a different approach, less based on semantics: the proposed pipeline consisted in a captioning module and a text-to-image synthesis model, which made it possible to learn the former on a small labelled dataset and use it to automatically annotate a large scale unlabelled dataset, which is then used to train the latter. By using the proposed pipeline, we provided evidence that it is possible to transfer labels from a smaller, curated dataset, to larger scale, unlabelled datasets, therefore making it possible to create new labels which follow the semantics of the unlabelled data. Finally, following the trail designed by the two previous chapters, in Chapter 3 we propose a method which uses both semantics and latent space representations to create new labels and examples: in fact, in the proposed technique a new latent representation of a caption (or video) is created by mixing in the representation of another caption which shares similar concepts at the semantic human-readable level. Moreover, by working in the latent space there are several advantages, including cheaper computations, the possibility to extend the same technique to multiple modalities, but also less concerns on privacy and data shareability, since only features are required to be shared, making it possible to apply these techniques to fields which have these issues.

In the second part of this Thesis, the focus is put on the customization of the training objective. We present several evidence that, by designing additional tasks or custom loss functions, it is possible to achieve better solutions by leveraging the available data to a greater extent and therefore without requiring additional annotations. In Chapter 5 we proposed a multi-task learning framework for video question answering. In particular, we designed this additional training objective by leveraging the structure of

the questions, which often involve common patterns, making them groupable. Therefore, by automatically adding a label to each of the questions, we were able to extend the typical learning framework, which is oriented on identifying the correct answer, to a more complex one which also predicts the type of the question, which may contain useful information to improve the training. In Chapters 6 and 7 we customize contrastive loss functions for the text-video retrieval task. In particular, in Chapter 6 we identify a fixed parameter in a contrastive loss function and redefine it in terms of the semantic overlap occurring between different captions. By doing so, at training time the similarity of the latent representation of different captions is defined in terms of how close they are; since the training is conducted contrastively, that is by optimizing such a similarity function, there is an introduction of semantic knowledge into the training process which greatly helps the generalization. In Chapter 7 the same semantic overlap is used to separate the captions into relevant or irrelevant with respect to a given query. By doing so, we address a shortcoming of the training objectives typically used for text-video retrieval, and also introduce a novel strategy used to discover new captions within the dataset which can be used to improve the training process. With these techniques, we achieved state-of-the-art results on multiple datasets, and also won a world-wide competition held at CVPR on multi-instance retrieval. Noteworthily, we obtained better results than a larger model which used more than 3 million video clips, compared to the 67 thousands that we used.

8.2 Future developments

- **On text-video retrieval**. Recent state-of-the-art methods for text-video retrieval [184, 257] learn to output a similar representation for both a video and its associated caption in a joint embedding space. This makes it possible to use a natural language query, mapped into the same embedding space, to retrieve and rank the videos in the dataset. In Chapters 6 and 7 we showed that this approach does not lead to high quality ranking lists, meaning that there are often many unrelated videos at the top of the ranking list, and that the introduction of semantic knowledge into the training process alleviates this problem. Nonetheless, all these approaches assume that the caption provided with a video captures all of its visual contents, yet this is often not the case because of the subjectivity and the presence of cultural biases in the captioning process: in fact, different human annotators may describe the same scene in different ways, for instance by using a diverse lexicon, by putting the attention on specific details, or by describing part of the background. Therefore, rethinking the methodology used to learn text-video retrieval models by putting less emphasis, and possibly removing the reliance, on the availability of a highly descriptive caption may represent an interesting direction.

- **On semantics and deeper understanding of concepts.** In the last few years, several concerns were raised about deep networks exploiting biases within datasets, language priors, and also visual priors [7, 119, 334]. As an example, in the VQA dataset [5] most of the binary questions, whose possible answers are either *yes* or *no*, can be answered correctly by choosing *yes* without even looking at the image [334]. These issues give a false impression on the progress in multimodal tasks,

and several authors addressed it by introducing new datasets, tasks, or metrics [86, 275, 298]. For instance, Thrush et al. [275] recently presented a novel task based on the matching of two images and captions containing the same set of words, only in a different order, and showed that most of the state-of-the-art vision-and-language models perform as badly as random chance. While the usage of multi-task learning may alleviate part of the problem, qualitative analyses may still show lack of understanding of higher level concepts and semantics. Considering that deep learning is being used to realize applications affecting millions of users across the globe, making sure the model is *understanding* what the user is asking is of utmost importance.

III
Appendix

9

Neural Turing Machines for the Remaining Useful Life Estimation problem

9.1 Introduction

One of the most important problems in the Prognostics and Health Management field is called Remaining Useful Life (RUL) estimation which consists in estimating how long it will take for a mechanical device under analysis to reach a situation where the likelihood of a failure is above a given threshold [121]. On the one hand, estimating the RUL precisely and reliably can have a great impact on maintenance-related costs, since it is possible to foresee when a failure will happen and thus plan accordingly the required intervention. On the other hand, failing such an estimation can have not only crucial economic consequences, but also a decrease in the reliability of a brand and may also create life-threatening situations, *e.g.* the disastrous crumbling of the I-35W in Minneapolis, Minnesota [114] or the more recent Morandi Bridge in Italy [197].

Common approaches for the RUL estimation can be divided into model-based and data-driven. Model-based approaches estimate the remaining life by leveraging mathematical or physical models of the degradation phenomena [274, 160]. These methods often require extensive expert knowledge, expensive verification, and for some components it is quite challenging to establish an accurate physical model. Differently from these approaches, data-driven methods rely on the availability of historical sensor data to build a degradation model. When it is not possible to observe multiple instances of each fault mode, for instance in industrial contexts where a fault may create life-threatening situations, these sensor data can be obtained via simulation software, *e.g.* in [245, 246], or experimental rigs, *e.g.* in [2, 260]. Many of the works following the data-driven approach manually design features in both the time and frequency domains and use them to learn a model through self-organizing maps [208], hidden Markov models [32], etc. While statistics play a major role in deciding which features to use and

how to combine them effectively, this feature engineering step can be time consuming and may still rely on prior knowledge. Conversely, deep learning makes it possible to automate the feature extraction process and work directly on the raw sensor data. The more successful deep learning-based approaches for this problem leverage sequence models to extract useful features from the sensor measurements and to identify temporal dependencies in the data. In particular, Long Short-Term Memory networks (LSTM) [103] are widely used as the key component to automate the feature extraction process [340, 284, 63, 305, 304].

An interesting neural network architecture which has not been explored until recently for the RUL estimation problem [68, 67] is the Neural Turing Machine (NTM). NTMs [88] are sequence models which, differently from LSTMs, interact with an external memory decoupled from the computation. As shown by Graves et al. [88], this makes it possible to achieve better performance on several algorithmic tasks, including the "associative recall", which resembles sequence modeling and consists in asking the model to recall an item from a list by querying it with the preceding item. This problem shares some similarities to the estimation of the RUL: if the list consisted of sensor measurements and associated RUL values, the current measurements could be used as a query to obtain the associated RUL value. Therefore, NTMs may also provide a more reliable tool than LSTMs for the RUL estimation problem.

In this Chapter, it is shown that even by using a simple model made of a single NTM and a decoder based on fully-connected layers it is possible to outperform widely adopted LSTM-based models, while also using fewer learnable parameters (28% less) and therefore with a smaller memory footprint. This is empirically validated with multiple experiments on two public datasets, the C-MAPSS dataset [245] and the PHM Society 2020 Data Challenge dataset [260]. Furthermore, the proposed method obtains competitive results with several state-of-the-art architectures which use deeper networks, bidirectional reasoning, or additional pretraining. Therefore, the experimental results show that providing access to an external memory can be beneficial to deal with the task, possibly implying that NTMs can be a better building block to design more complex architectures and to automatically extract features from the available time series.

The major contributions of this Chapter can be summarized as follows:

- A thorough empirical study is performed to explore the usage of NTMs as the main feature extraction component for the Remaining Useful Life estimation problem.
- Consistent empirical evidences are provided to show that NTMs are a powerful and efficient model which outperforms the more popular LSTM-based models, while being also less parameter-demanding and thus more memory-efficient which makes it possible to use in memory-constrained environments. Given the similar underlying nature of the two sequence models, it is possible to integrate the NTM within other architectures and improve the final performance.
- The proposed simple model achieves an estimation error comparable and even competitive with respect to the error obtained by other architectures found in literature, which are more complex, use ensemble of models, and additional pretraining.
- Multiple experiments are performed on two public datasets dealing with aircraft

engines (C-MAPSS) and particle filtration systems (PHM Society 2020 Data Challenge), showing the strengths of the proposed model while also highlighting some limitations related to industrial contexts.

- The source code is released, to ensure reproducibility and to provide a strong, open source baseline otherwise difficultly found in the community.

In Section 9.2 the scientific literature related to this Chapter is introduced. Then, in Section 9.3 the details about the methodology are described, by focusing on the application of the Neural Turing Machine to the RUL estimation task. All the experiments performed throughout this Chapter are described in Section 9.4. Finally, in Section 9.5 some conclusions are drawn about the work done and possible future works are mentioned.

9.2 Related Work

Model-based and data-driven methods. Estimating the remaining useful life of a system has been a strategic research problem for several decades [22, 89]. Traditional approaches can be divided into model-based and data-driven.

The former are based on the availability of mathematical or physical models of the degradation phenomena, such as spall propagation models for rolling bearing elements [17] or crack growth models for a system experiencing fatigue [37]. Model-based methods require an in-depth understanding of the underlying system and the failure modes. Furthermore, these approaches are built in a case-by-case scenario, making them difficult to be applied in different contexts without spending a considerable effort.

Methods following a data-driven approach rely on the availability of historical sensor data to build a degradation model. In situations where the machinery is frequently maintained and faults are never observed, the sensor data can be obtained through simulation models [245, 246] or through experimental rigs [2, 260]. Notable examples of data-driven methods include statistical methods, such as Auto Regressive Integrated Moving Average [206, 345] and hidden Markov models [32, 192], and Artificial Intelligence methods, *e.g.* by using self-organizing maps [208] and Support Vector Machines [112]. Differently from the model-based approach, the data-driven methodology does not require a deep understanding of the underlying system. Yet, in data-driven methods which do not employ deep learning the features are manually designed and extracted from the raw data, therefore this feature engineering step can be time consuming and may still rely on domain knowledge.

Deep Learning-based methods. Differently from previous approaches, deep learning makes it possible to automate the process of feature extraction from the raw sensor data. The attention towards these techniques has been promoted thanks to the availability of public datasets (*e.g.* [245, 2]) and the possibility to exploit high quality sensors to frequently measure the evolution of different characteristics of the mechanical system under analysis.

Basing their works over the assumption that time series can be interpreted as images and the success obtained in computer vision tasks, approaches based on Convolutional Neural Networks (CNN) were explored for RUL estimation. Babu et al. [9] and Li et al. [162] explored deep CNN-based methods, whereas Cornelius et al. [38] leverages

heteroscedastic and epistemic uncertainties to improve the RUL estimation in a deep CNN-based network. Nonetheless, the temporal dependencies occurring in sensor data hardly are learned by these techniques.

Therefore, sequence models (*e.g.* LSTM networks) are often exploited because of their ability to model the evolution of the measured features. Both a Multilayer Perceptron and a RNN were used in [99]. Due to the length of the time series considered in this field, RNNs can have problems remembering the important information and capturing long term dependencies, which has encouraged several researchers to exploit memory-based networks to store key information, such as GRUs [13, 283, 181] or LSTMs [340, 284, 63, 304, 13, 329]. Recently, [305] proposed an ensemble of bidirectional LSTM-based models, each trained on the input data framed with windows of a different size, in order to have each model focus on temporal dependencies that require more or less time to develop. A sequence model which was only recently used in some works [68, 67] on RUL estimation consists in the NTM. Graves et al. introduced this sequence model in [88], where they show that NTMs outperform LSTM networks in five tasks with increasing complexity, including: the "copy" task, which requires the model to observe a sequence and copy it in output; the "repeated copy", which is an extension of the previous task and asks the model to repeat the copy a given number of times; the "associative recall" task, which resembles sequence modeling and consists in recalling an input item that follows a given "query" item in a previously processed list of items; the "dynamic n-gram", which tests the capabilities of the models to learn a probability distribution; finally, the "priority sort" task, which requires the model to learn how to sort a sequence based on a priority vector. NTMs were introduced within the RUL estimation field by Falcon et al. in [68] and [67], where they obtained a lower prediction error than previously published works. Yet, in these works the NTM is used within bigger and complex architectures, so it is not clear how much the NTM is helping the whole approach. Conversely, in this Chapter the attention is driven towards the contribution given by the NTM when used as the whole feature extraction mechanism. In particular, even by using a single NTM as the feature extractor better performance are achieved than more popular LSTM-based methods, while also using around 28% fewer learnable parameters. Furthermore, these results are empirically validated by performing multiple experiments on two public datasets, the C-MAPSS dataset [245] and the PHM Society 2020 Data Challenge dataset [260].

9.3 The proposed Approach

A schematic overview of the proposed approach is shown in Fig. 9.1. First of all, the raw input time series are preprocessed following three simple steps: they are normalized using MinMax, *i.e.* $x_i' = \frac{2(x_i - X_{min})}{X_{max} - X_{min}} - 1$ where X_{max}, X_{min} represent maximum and minimum value of feature X; labeled, during training, with a piece-wise linear degradation function [99], limiting the RUL to 125; and cut into shorter time series using a sliding window technique. After the preprocessing, they are fed to the cells of the NTM. The hidden states computed by the NTM are interpreted as the feature vectors for the given time series. Each feature vector is then mapped to estimate RUL values through a simple decoder made of two stacked fully connected layers.

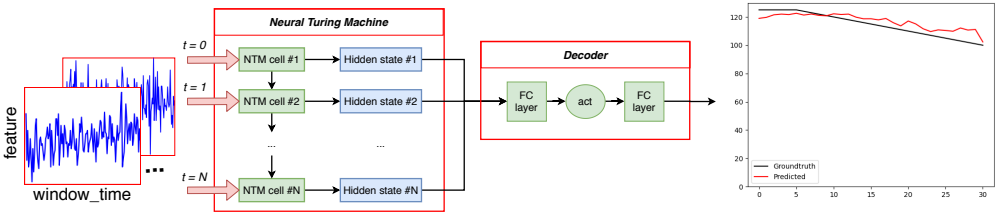


Figure 9.1: A graphical overview of the proposed approach. The time series are first cut into shorter windows, then fed to the network. The Neural Turing Machine is used as the feature extractor. Finally, two stacked fully connected networks are used to map the extracted features to a sequence of RUL values.

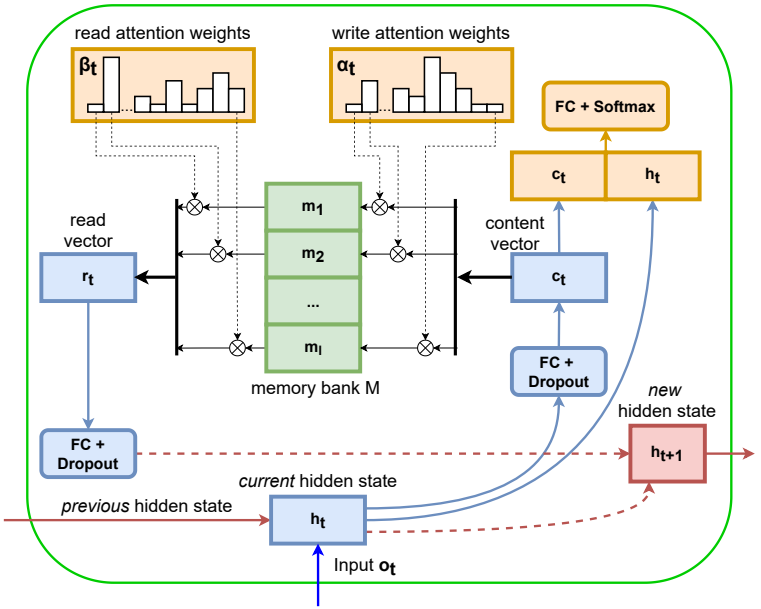


Figure 9.2: At time t, the NTM updates the hidden state h_{t+1} (red) by using its memory M, input vector o_t, and previous hidden state h_t. It is functionally separated into attention (yellow) with weights α_t and β_t, read/write operations (blue), and memory slots (green).

9.3.1 Neural Turing Machine

In the NTM, shown in Fig. 9.2, the memory bank $M \in \mathbb{R}^{l \times s}$ is made of l memory locations, *i.e.* $M = [m_1, m_2, ..., m_l]$, where each of them is a vector with s features, *i.e.* $m_i \in \mathbb{R}^{1 \times s}$. Considering as input a windowed time series $T \in \mathbb{R}^{t_l \times f}$, made of t_l vectors of size f, the NTM sequentially processes T by extracting, storing, and eventually retrieving some of the most important information from each of the t_l measurement vectors using learnable read and write operations. In this way, the operations performed

by the NTM use many memory vectors, whereas LSTM networks rely on one memory vector and thus it is likelier that previously obtained information is rewritten and lost. The NTM also updates an hidden state $h_t \in \mathbb{R}^{1\times s}$ at each time step, which acts as a summary of the measurement vectors received so far and their interactions over time. The sequence of hidden states $H = \{h_i \,|\, i \in \mathbb{N}^+, i \leq t_l\}$ is then used as the automatically extracted features of T. Hence, the feature extraction process depends entirely on the NTM and its interactions with the raw sensor data, whereas Falcon et al. used the NTM on top of an LSTM [67] or in conjunction with a CNN and self-attention layers [68], therefore making unclear the contribution given by the NTM in the whole approach. As shown in Fig. 9.2 the input to the NTM at time step t is represented by $o_t \in \mathbb{R}^{1\times f}$ and consists of the measured value of f different sensors. Three types of operation are sequentially performed by the NTM: write, read, and hidden state update.

Write operation. At time t, the memory bank is updated by writing new information obtained by the current sensor measurements o_t and the previous hidden state h_{t-1}. In this way, it stores new knowledge from the input, while maintaining some of the previously captured information. The information to be written into the memory, *i.e.* the content vector $c_t \in \mathbb{R}^{1\times s}$, is computed as follows:

$$c_t = \delta_{p_1}(\sigma([o_t, h_{t-1}]W_{hc} + b_c)) \tag{9.1}$$

where δ_{p_1} is the dropout [266] operator with probability $p_1 \in [0, 1]$, $[\cdot]$ is the concatenation operator, $o_t \in \mathbb{R}^{1\times f}$ represents the current sensor measurements vector, and $h_{t-1} \in \mathbb{R}^{1\times s}$ the previous hidden state. $W_{hc} \in \mathbb{R}^{(f+s)\times s}$ and $b_c \in \mathbb{R}^{1\times s}$ are trainable parameters. To determine how much the memory should be modified, an attention mechanism is used in order to compute a weight for each of the l memory locations. This is done as following:

$$a_t = \delta_{p_1}(v_a \; tanh([c_t, h_{t-1}]W_{ha} + b_a)) \tag{9.2}$$

$$\alpha_{t,i} = \frac{exp(a_{t,i})}{\sum_{j=1}^{l} exp(a_{t,j})} \text{ for } i = 1, ..., l \tag{9.3}$$

In particular, $\alpha_t = \{\alpha_{t,1}, \ldots, \alpha_{t,i}, \ldots, \alpha_{t,l}\}$ represents the attention weights and Eq. 9.3 satisfies $\sum_i \alpha_{t,i} = 1$. $W_{ha} \in \mathbb{R}^{(s+s)\times l}$, $v_a \in \mathbb{R}$, and $b_a \in \mathbb{R}^{1\times l}$ are trainable parameters. Differently from previously published NTM-based networks, the dropout operator is added both in Eq. 9.1 and Eq. 9.2 in order to mitigate overfitting situations and to improve the generalizability. Each memory slot m_i is then updated in the following way:

$$m_i = \alpha_{t,i} c_t + (1 - \alpha_{t,i}) m_i \text{ for } i = 1, ..., l \tag{9.4}$$

using the attention weights to balance how much of the new information (in c_t) should be saved inside the memory. This also helps the model to understand the underlying relations between the evolution of the different sensors: the new information is spread unevenly throughout all the memory locations, making it possible to store in each of them different key information.

Read Operation. After the update of the memory bank M, the NTM reads from M the new data which is used to update the hidden state. This operation is performed in a similar way to the write operation. First of all, as in Eq. 9.2 and Eq. 9.3, attention

weights are computed on c_t and h_{t-1}. As for the write operation, the computation of the attention is regularized by the dropout operator. Then, the attention weights are used to compute a weighted average of the vectors currently contained in the memory, obtaining a read vector r_t. Since M contains information gathered from all the sensor measurements and their interaction over time, r_t represents an informative summary of the current health-related state of the mechanical system.

Dropout-augmented read and write operations. As mentioned before, both the write and read operations are augmented in this work by adding the dropout operator. As shown by Srivastava et al., this operator may mitigate overfitting and improve the performance on many heterogeneous tasks [266]. This is possible by reducing the co-adaptation: at training time, the parameters of the neurons are updated in a way such that they try to fix the mistakes made by other neurons, i.e. they co-adapt. By using the dropout, a fraction of the neurons is randomly shut off at training time, therefore making the neurons less prone to rely on co-adaptation. Since this phenomenon is unlikely to generalize, reducing it may lead to improved generalization.

Hidden State Update. Finally, the hidden state is updated with the knowledge gathered from the updated memory bank, the previous hidden state, and the current input measurements:

$$h_t = \sigma(o_t W_{oh} + r_t W_{rh} + h_{t-1} W_{hh} + b_h) \tag{9.5}$$

where $W_{oh} \in \mathbb{R}^{f \times s}$, $W_{rh} \in \mathbb{R}^{s \times s}$, $W_{hh} \in \mathbb{R}^{s \times s}$, and $b_h \in \mathbb{R}^s$ are trainable parameters. The sequence of hidden states $h_1, h_2, \ldots, h_N$ are grouped in a matrix $H \in \mathbb{R}^{t_l \times s}$ and used as the automatically extracted features for the input time series.

9.3.2 RUL Decoder

After the features are extracted by the NTM, a simple decoder (see Fig. 9.1) is employed to learn a mapping from these features to RUL values:

$$d_1 = \delta_{p_2}(\sigma(H W_{d_h} + b_{d_h})) \tag{9.6}$$

$$d_2 = d_1 W_{d_o} + b_{d_o} \tag{9.7}$$

where $W_{d_h} \in \mathbb{R}^{s \times fc}$, $W_{d_o} \in \mathbb{R}^{fc \times 1}$, $b_{d_h} \in \mathbb{R}^{fc}$, and $b_{d_o} \in \mathbb{R}$ are trainable parameters, and σ is the sigmoid function. As the output of this step, $d_2 \in \mathbb{R}^{t_l \times 1}$ is obtained, which represents the sequence of predicted RUL values.

9.3.3 Loss function

To train the model and obtain optimal weights and biases from a labelled and preprocessed training dataset, the Mean Square Error (MSE) of the predicted RUL is optimized with respect to the groundtruth values. It is defined as: $MSE = \frac{1}{n} \sum_{i=1}^{n} (RUL'_i - RUL_i)^2$, where n is the total number of data samples, RUL'_i and RUL_i represent respectively the predicted and groundtruth RUL for the i-th data point.

Description of the characteristic	Units
Total temperature at fan inlet	°R
Total temperature at LPC outlet	°R
Total temperature at HPC outlet	°R
Total temperature at LPT outlet	°R
Pressure at fan inlet	psia
Total pressure in bypass-duct	psia
Total pressure at HPC outlet	psia
Physical fan speed	rpm
Physical core speed	rpm
Engine pressure ratio	–
Static pressure at HPC outlet	psia
Ratio of fuel flow to static pressure at HPC outlet	pps/psi
Corrected fan speed	rpm
Corrected core speed	rpm
Bypass ratio	–
Burner fuel-air ratio	–
Bleed enthalpy	–
Demanded fan speed	rpm
Demanded corrected fan speed	rpm
HPC coolant bleed	lbm/s
LPT coolant bleed	lbm/s

Table 9.1: Description of the 21 sensors available in the C-MAPSS dataset, from [246].

9.4 Experimental Results

9.4.1 Datasets under analysis

To evaluate the proposed methodology, two public datasets are considered: the NASA C-MAPSS Turbofan Engine Degradation Simulation Dataset [245] and the PHM Society 2020 Data Challenge [260].

The **C-MAPSS** dataset consists of 4 subdatasets (named FD001, FD002, FD003, and FD004) of multiple multivariate time series. Every measurement vector contained in each series is made of three operational settings and 21 sensor values, each recording distinct physical characteristics of the considered system. These include temperature and pressure measured at the fan inlet, and the temperature measured for each module in the gas path (HPC, HPT, and LPT). A full list of the considered characteristics can be found in Table 9.1 (from Saxena et al. [246]), whereas Figure 9.3 (left) presents a simplified diagram of the turbofan engine. The data points come from different turbofan engines of the same type, which have different levels of initial wear. While training series are run-to-failure, testing series have a positive RUL which represents the target label. Table 9.2 shows a summary of the number of time series, fault and operational conditions found in each subdataset.

In the experiments performed in this Chapter, some of the raw input features are ignored because they are constant and thus not informative. In particular, 14 sensor measurements out of the total 21 sensors are kept, whose indices are 2-4, 7-9, 11-15, 17, and 20-21. A similar selection is also done in [63, 67, 62].

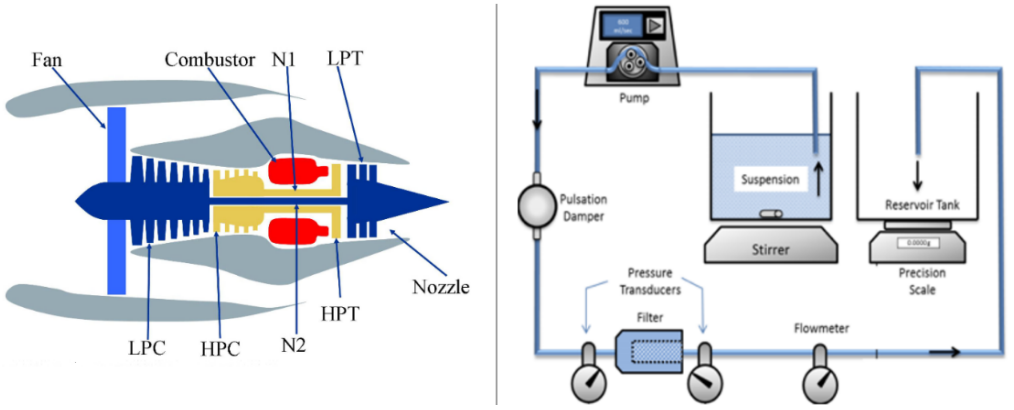


Figure 9.3: (left) Diagram of the turbofan engines considered for the C-MAPSS dataset [246]. (right) Diagram of the experimental rig used for the PHM Society 2020 challenge [260].

Moreover, the three operational settings can be used in datasets FD002 and FD004 to identify six operational conditions, as reported in [245]. These settings affect the measurements because an engine behaves differently based on its operational condition (*e.g.* whether the airplane is taking off or it is cruising above the clouds). Hence, for these two datasets, KMeans [185] is used to cluster the measurements into six groups. Then, the data within each cluster are normalized using MinMax, in order to use the same scale to treat data sampled in the same condition [9]. Each of the measurement vectors in FD002 and FD004 is then augmented concatenating the one-hot encoding of the operational condition, thus increasing the input size from 14 to 20. A similar data preprocessing approach was also reported in [340, 68, 9].

Finally, a value t_l for the window size is decided and used over all the four datasets. To determine t_l, five different values are tested: 30, 40, 50, 60, and 70. For easier reproducibility, Table 9.2 reports the amount of windowed time series observed during training after the train/validation split.

The second dataset, abbreviated to "**PHM20**", is released as part of the PHM Society 2020 Data Challenge [260] and consists of sensor measurements collected from an experimental rig used to simulate failures in a particle filtration system, which are widely used in industrial environments. This type of system is subject to clogging due to the presence of contaminants in the liquids and, in this case study, such a clog can be identified when the pressure difference is higher than 20 psi. The public dataset consists of 24 experiments for training and 8 for validation. Each experiment is annotated with the concentration (from 40% up to 47.5% with 2.5% increments) and the size of the particles (in the range 45-53um, or in the range 63-75um), and consists of several thousands of measurements, sampled at 10 Hz. Each sample is annotated with three sensors: the flow rate measured with a flowmeter, and both the upstream and downstream pressures which are measured with pressure transducers. A schematic of the experimental rig is shown in Fig. 9.3 (right). For each time step, five input features are considered: the

Dataset	FD001	FD002	FD003	FD004
Train time series	100	260	100	248
Test time series	100	259	100	248
Operating conditions	1	6	1	6
Fault conditions	1	1	2	2
Max length (testing)	303	367	475	486
Min length (testing)	31	21	38	19
Training samples with				
t_l=30	12100	32756	15499	38350
t_l=40	11400	30936	14799	36610
t_l=50	10700	29116	14099	34870
t_l=60	10000	27296	13399	33130
t_l=70	9300	25476	12699	31390

Table 9.2: Summary of the subdatasets of the C-MAPSS dataset.

three sensors, the concentration value, and the size of the particles. Then, these values are normalized with MinMax. In this Chapter, the RUL is considered to be 0 when the pressure difference is higher than 20 psi and the time series are labelled with the piece-wise degradation function (with 125 as the maximum value), as in Ince et al. [115]. As in the previous case, five values for the window size t_l are considered but, since the time series in PHM20 are longer than those in C-MAPSS, these values are bigger than before: 70, 140, 210, 280, and 350.

9.4.2 Model evaluation

To evaluate the performance, the main metric used consists in the Root Mean Square Error (RMSE). Furthermore, for the C-MAPSS dataset the Scoring Function is also considered, whereas the Mean Absolute Error (MAE) is used for the PHM20 dataset.

Scoring Function

The Scoring Function was initially proposed in [245] and is defined as:

$$S = \sum_{i=1}^{n} s_i \text{ , where } s_i = \begin{cases} e^{\frac{-e_i}{13}} - 1, e_i < 0 \\ e^{\frac{e_i}{10}} - 1, e_i \geq 0 \end{cases} \tag{9.8}$$

where S is the computed score, n is the total number of data samples, and $e_i = RUL'_i - RUL_i$ is the difference between estimated and groundtruth RUL at the i-th data point. The asymmetric nature of this function penalizes more the "too optimistic"

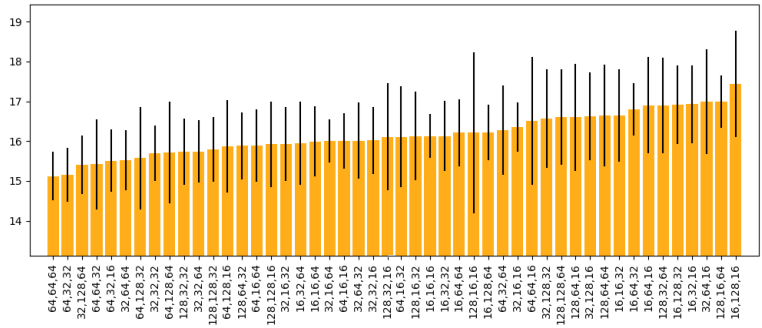


Figure 9.4: Average testing RMSE (with standard deviation) on FD001 using the best validation model. Hyperparameters s, l, and fc shown on x axis.

predictions, meaning that it gives a higher score (*i.e.* worse) when the model predicts $\hat{v}$ but the true RUL is v such that $v < \hat{v}$, thus leading to an unpredicted early failure of the considered system. On the other hand, it penalizes less an "early" prediction (*i.e.* $\hat{v} < v$): although such prediction may trigger a superfluous maintenance, it should not lead to unexpected failures, possibly avoiding more severe consequences.

Root Mean Square Error and Mean Absolute Error

RMSE and MAE are commonly used to evaluate prediction accuracy, both giving equal weights for both early and late predictions. A key difference between the two metrics is how much they punish observations which are further from the mean: in particular, RMSE is a quadratic scoring rule, therefore it is more sensitive to large errors in the predictions.

9.4.3 Implementation details

The training is performed for 50 epochs, using a variable learning rate, starting from 0.005 and decaying it by a factor of 0.6 every 15 epochs. For C-MAPSS, the training time series (before cutting them into windows) are split with a 70/30 ratio to create the training and validation splits. For the PHM20 dataset, the original validation set is used as the testing set, and the training set is split with a 80/20 ratio to define the train and validation splits. 10 runs are performed and for each the best model on the validation set is selected and used for testing. During training, the mini-batch gradient descent (batch size 100), and RMSProp (momentum 0.9, weight decay 0.0005) are employed. The dropout rates are set to $p_1 = 0.1$ and $p_2 = 0.25$.

The bias b_h is initialized to zero, and the weights v_a and v_b are sampled from a normal distribution with mean 0 and standard deviation 0.01. All the other weights and biases are initialized by sampling from a uniform distribution in the range $[-\sqrt{k}, \sqrt{k}]$, where $k = \frac{1}{in\_fts}$ and in_fts is the number of input features of the weight or bias (*e.g.* for $W_{hb} \in \mathbb{R}^{(s+s)\times l}$, $k = \frac{1}{s+s}$). The memory bank and hidden state are initialized with zeros.

Method	s, l, fc values	Param. count	RMSE
LSTM [340]	32, 64, 8	31,761	16.14
Our NTM	32, 128, 64	22,945	$15.23_{\pm 0.66}$
LSTM	32, 48, 8	23,057	16.40
Our NTM	64, 64, 64	35,105	15.46

Table 9.3: Comparison of RMSE over FD001 with respect to the LSTM-based solution proposed in [340]. Better results, more efficiently (about 28% fewer parameters).

Finally, PyTorch 1.3.0 is used to implement the proposed solution[1].

9.4.4 Discussion of the results on the CMAPSS dataset

Grid search results. To determine the best combination of the hyperparameters used in the proposed approach, *i.e.* the two sizes s and l of the NTM, and the hidden size, fc, of the decoder, multiple runs are performed on FD001. By using $fc = 8$, high RMSE values (around 12% higher than other combinations) are obtained, possibly implying that such a low number of neurons in the decoder is not enough to learn a meaningful RUL estimation function. Higher values for fc lead to better accuracy, although s and l influence the overall performance as well, as shown in Fig. 9.4.

Parameter efficiency of the NTM. By using the external memory, the NTM may learn better features with less parameters. To investigate this, a fair comparison to the LSTM-based solution proposed in [340] is made, because they perform hyperparameter optimization as well. Table 9.3 reports a lower estimation error for the NTM (15.23 compared to 16.14), while also using 28% fewer parameters (22945 compared to 31761), making it a better approach when dealing with memory-constrained environments, *e.g.* embedded. Furthermore, even if the two networks had a similar amount of parameters, the NTM would still perform better, as reported in Table 9.3 (15.46 compared to 16.14 using around 32000 parameters, and 15.23 to 16.40 using 23000 parameters).

Window sizes. The size of the windows used during training (see Sec. 9.3) can be seen as another hyperparameter, since longer series may be more difficult to deal with but may also be more informative. Fig. 9.5 reports the error obtained over the four datasets using the five best combinations of hyperparameters (s, l, fc) found in the previous experiment, and five different sizes for the windows, *i.e.* 30, 40, 50, 60, and 70. For dataset FD001 (Fig. 9.5.a), short windows are more beneficial than longer time windows. The easier nature of this dataset may explain this, since the RUL values are influenced only by one operating condition and one fault condition. Conversely, for FD002 to FD004 these short time windows are not optimal, as the events which lead to a fault likely require more time to develop. All the following experiments use 70 as the window size.

Qualitative analysis. Figure 9.6 displays the predictions (blue) and groundtruth values (red) over the four datasets, showing that the proposed model can effectively estimate the RUL of unseen time series in the testing set. To give further evidence of this, Fig. 9.7 presents some examples of prediction on the testing set, showing that

[1]The code is released at: https://github.com/aranciokov/NTM-For-RULEstimation

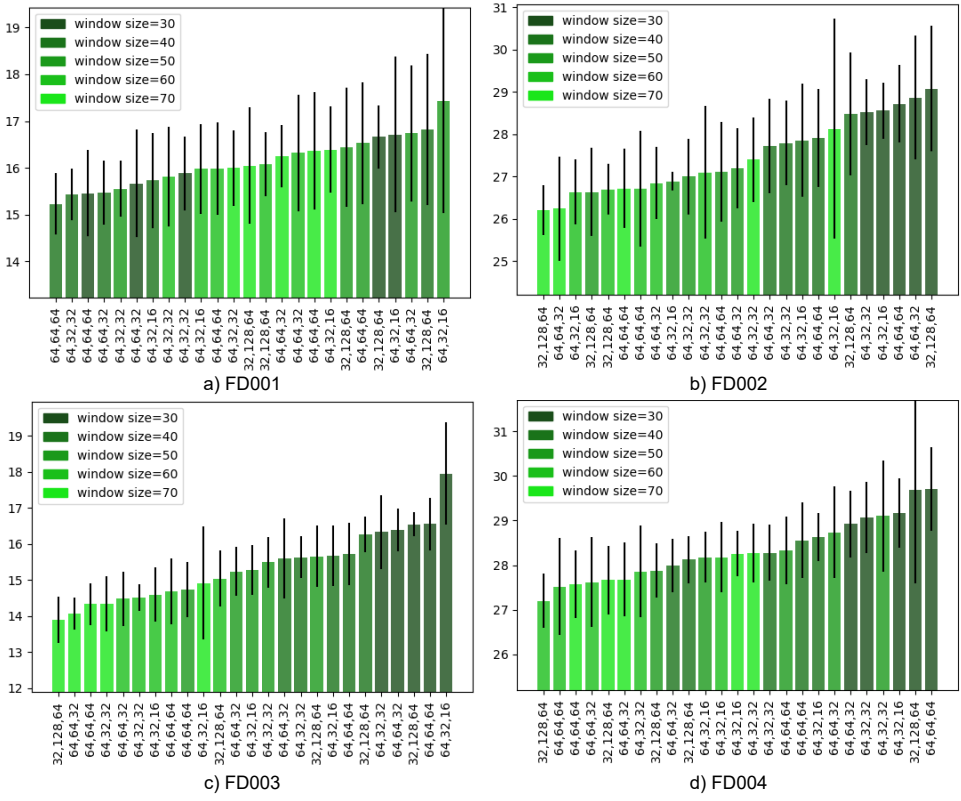


Figure 9.5: Average testing RMSE (with standard deviation) using five combinations of s, l, and fc (see Sec. 9.4.4), and five values for the window size. Over FD001, shorter time windows lead to better results, whereas longer windows are preferred for the other, more complex datasets. Best viewed in color.

the RUL progression can be reliably predicted. Moreover, the accuracy increases as the fault gets closer.

Limitations of the piece-wise function. In the experiments presented both in this Chapter and in the literature, *e.g.* by Zheng et al. [340] and Babu et al. [9], the estimation error observed on FD002 and FD004 is higher than FD001 and FD003. This is mainly due to two factors. First, datasets FD002 and FD004 are more difficult, due to multiple operating conditions affecting the captured measurements. Secondly, the de facto standard degradation function [99] used to label the C-MAPSS dataset is not a perfect solution. In fact, since it limits the maximum RUL value observed during training, it is hard for any model to correctly predict at testing time RUL values which are higher than such a maximum. As an example, FD004 contains 67 (out of 248) times series with a RUL higher than 125 and the model fails these predictions (see Fig. 9.6.d).

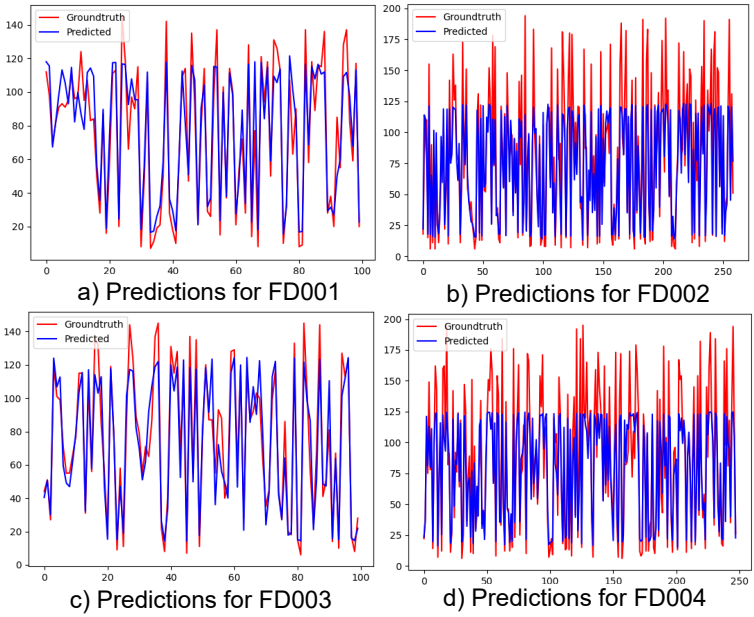


Figure 9.6: Comparison between predicted (blue) and groundtruth RUL values (red).

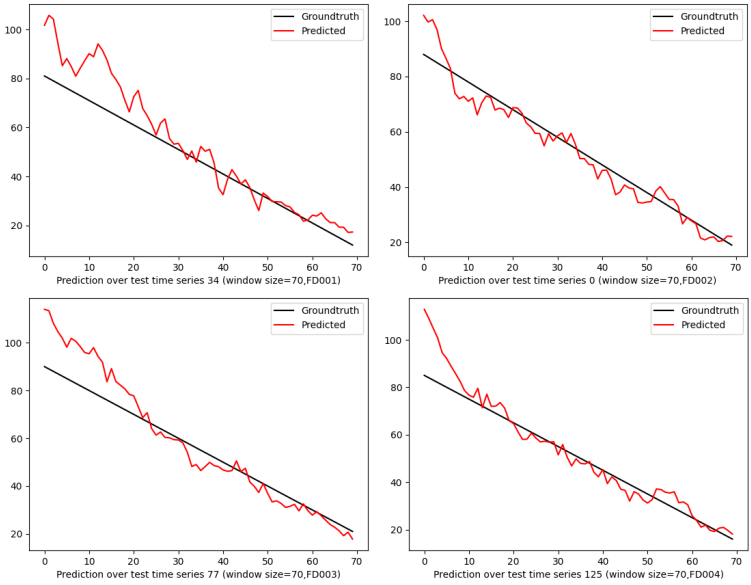


Figure 9.7: Examples of prediction (red) made by the proposed model during testing. The model generalizes well over unseen test examples. Note that the true RUL for testing series is only known for the last time step: a linear degradation is shown here for comparison.

Methods	FD001	FD002	FD003	FD004
MLP [9]	1.8×10^4	7.8×10^6	1.7×10^4	5.6×10^6
CNN [9]	1.3×10^3	1.4×10^4	1.6×10^3	7.9×10^3
LSTM [340]	$\mathbf{3.4\times10^2}$	$\mathbf{4.4\times10^3}$	8.5×10^2	5.5×10^3
Our NTM	3.8×10^2	5.5×10^3	$\mathbf{3.0\times10^2}$	$\mathbf{5.2\times10^3}$
BiLSTM [284]	2.9×10^2	4.1×10^3	3.2×10^2	5.4×10^3
GADLM [62]	2.3×10^2	3.4×10^3	2.5×10^2	2.8×10^3
NTM-Hybrid [68]	2.1×10^2	6.0×10^3	2.7×10^2	4.8×10^3

Table 9.4: Comparison on the C-MAPSS dataset using the Scoring function (see Sec. 9.4.4 for the discussion).

Methods	FD001	FD002	FD003	FD004
MLP [9]	37.56	80.03	37.39	77.37
CNN [9]	18.45	30.29	19.82	29.16
LSTM [340]	16.14	**24.49**	16.18	28.17
Our NTM	$\mathbf{16.05_{\pm1.2}}$	$26.21_{\pm0.6}$	$\mathbf{13.90_{\pm0.6}}$	$\mathbf{27.67_{\pm0.7}}$
DCNN [162]	13.32	24.86	14.02	29.44
BiLSTM [284]	13.65	23.18	13.74	24.86
GADLM [62]	12.56	22.73	12.10	22.66
Sim-Sup [108]	18.33	-	12.73	-
MTW-BLSTM [305]	12.61	-	-	-
Multi-Local [183]	14.1	-	-	-
NTM-Hybrid [68]	12.53	27.04	13.73	28.11

Table 9.5: Comparison on the C-MAPSS dataset using the RMSE (see Sec. 9.4.4 for the discussion).

A new challenge is thus brought to light: the function proposed in [99] is in fact widely accepted and used [340, 284, 305, 68, 9, 162], yet this study shows it may not be the most appropriate solution for the task. Few works are actively working on alternative labeling functions, *e.g.* [63], but no definitive solutions are available to date.

Comparison with similar architectures. In Tables 9.4 and 9.5, the results obtained using $s = 32$, $l = 128$, $fc = 64$ are presented and compared to other published works. In particular, since the focus of this Chapter is to explore the NTM as the main feature extraction component, it is more fair to compare to architectures following similar principles. As shown in both the Tables, models leveraging the sequential nature of the data, i.e. LSTM and NTM, obtain a lower estimation error. Moreover, the NTM excels in both metrics on the datasets involving multiple fault conditions (FD003 and FD004), and in terms of RMSE on FD001 (Table 9.5).

Comparison with more complex architectures. For a more comprehensive comparison, published works with more complex architectures, pretraining, *etc* are also considered in Tables 9.4 and 9.5. Compared to the deep CNN used by Li et al. [162], the NTM achieves a lower RMSE when multiple fault conditions affect the sensor measurements (13.90 and 27.67 compared to 14.02 and 29.44). Interestingly, although using only one direction to analyze the data, it achieves lower scores (3.0×10^2 and 5.2×10^3

s	l	fc	RMSE
16	128	64	11.35
32	128	64	7.03
64	128	64	5.87
128	128	64	7.10
64	16	64	10.11
64	32	64	5.50
64	**64**	64	5.45
64	128	64	5.87
64	64	16	16.46
64	64	32	7.32
64	64	**64**	5.45
64	64	128	6.12
64	**64**	**64**	**5.45**

Table 9.6: Hyperparameters optimization on PHM20 dataset.

compared to 3.2×10^2 and 5.4×10^3) than bidirectional LSTMs [284]. Multiple models (*e.g.* Xia et al. [305]) and additional pretraining (*e.g.* Ellefsen et al. [62]) lead to more accurate predictions, but these techniques could be also applied to NTM-based solutions. Moreover, all the LSTM-based solutions could be further improved by replacing the LSTM with an NTM.

Finally, there are other recent papers employing LSTM (*e.g.* [304]) or CNN components (*e.g.* [172]) which perform better. Yet, a fair comparison is difficult to make: their experimental setting is different as the testing labels are also rectified by the piece-wise function, which is not done in this Chapter.

9.4.5 Discussion of the results on the PHM20 dataset

Hyperparameters tuning. As for the C-MAPSS dataset, the influence of the hyperparameters of the NTM (s, l, and fc) is explored on the PHM20 dataset. Since it contains longer time series, the window size t_l is initially fixed to 210. Here, s, l, and fc are varied in {16, 32, 64, 128}. The average RMSE (on 10 runs) is reported in Table 9.6. Two observations can be made. First of all, 64 represents an optimal value among those analyzed for the three hyperparameters, leading to an RMSE of 5.45. Secondly, the size s of the memory locations and the number fc of neurons in the decoder are highly influential on the final performance.

Temporal context and external memory. By continuously interacting with the external memory, NTMs may be able to deal with longer time series than LSTMs. This may be strategic in industrial settings, where sensor measurements can be collected frequently over long periods of time. To confirm this surmise, the performance of the two networks are compared as the length of the temporal context t_l increases. For the LSTM the same hyperparameters as in Zheng et al. [340] are used, whereas for the NTM $s = 64$, $l = 64$, and $fc = 64$. Table 9.7 reports the performance both in terms of MAE and RMSE. For the NTM, $t_l = 280$ represents an optimal value, whereas for the LSTM $t_l = 70$ leads to best results, although the error is far higher than the one achieved with the NTM. Two observations can be made. Firstly, the surmise is confirmed, since the

	Our NTM		LSTM [340]	
window size t_l	MAE	RMSE	MAE	RMSE
70	4.44	6.82	4.97	7.05
140	4.54	6.74	5.38	7.77
210	3.74	5.45	5.09	7.30
280	**3.73**	**5.37**	5.45	7.59
350	4.97	6.93	5.45	8.42

Table 9.7: Comparison on PHM20 dataset between the proposed model and the LSTM-based model from [340]. Performance is measured both with RMSE and MAE.

	MAE	RMSE
Random Forest	3.97	7.31
Gradient Boosting	3.82	6.81
LSTM	4.97	7.05
Our NTM	**3.73**	**5.37**

Table 9.8: Comparison with state-of-the-art methods on PHM20 dataset, including the first two methods from [115] and the results obtained by the LSTM-based model (based on [340]).

NTM manages to deal with longer sequences, whereas the LSTM shows a decreasing accuracy as the sequences become longer. Secondly, the NTM has a better prediction capability than the LSTM on all the values tested for t_l, obtaining an estimation error as low as 5.37 RMSE and 3.73 MAE compared to 7.05 RMSE and 4.97 MAE obtained by the LSTM.

Comparison with state-of-the-art. Table 9.8 reports a comparison to state-of-the-art methods which took part into the PHM Society 2020 Data Challenge [260], alongside the LSTM-based model used in previous experiments. The winner of the challenge (Lomowski et al. [177]) used a non-comparable methodology, therefore it is not included here. Ince et al. [115] used Machine Learning techniques, including random forest and gradient boosting (implemented with Scikit-learn [213] and CatBoost [219]). For these two methods, the results obtained by running the public implementation provided by the authors are reported. With the proposed approach a lower estimation error is obtained, measured both with MAE and RMSE.

9.4.6 Discussion of NTM applicability to industrial contexts

The previous subsections show that a more accurate prediction is achieved if the NTM is used to automatically extract the features from the input series. In particular, on both the C-MAPSS and the PHM20 dataset a lower estimation error is observed when compared to a popular LSTM-based model. Nonetheless, a superior accuracy may not be the only factor which needs to be taken into account when implementing a RUL estimation system.

Training times and size of the dataset. In a scenario in which the historical data form a sizable dataset, the NTM may require a bigger time investment to perform the training. As an example, on a system with a NVIDIA RTX A5000, an i7-9700K,

and 32GB of RAM, training the NTM on the PHM20 dataset takes around 1 hour (single run), whereas it takes 10 minutes for the LSTM-based model. This is mainly due to the availability of a CUDNN implementation for the LSTM, which uses low level routines to reduce the running time of each operation. Conversely, the implementation of the NTM relies on higher level tools, making it slower. Nonetheless, the NTM and the LSTM share similar primitives, therefore a CUDNN implementation for the NTM is theoretically possible and may reduce the gap. With the current implementation, learning from very big datasets by means of a NTM-based system may become far too resource-consuming.

Low latency scenario. Industrial systems may need to predict the RUL with a negligible latency. In a similar scenario, the NTM may not be optimal. In fact, on the same system as before, the NTM estimates the RUL in around 55 ms (tested on 200 sequences of length 70), whereas the LSTM takes around 3 ms. As before, this is due to the usage of high level tools for the NTM, and it may be alleviated by changing implementation. Nonetheless, at the current state, if the RUL estimation system needs to perform a prediction with a really low delay (near real-time), then a LSTM-based implementation is preferable although with an inferior accuracy.

9.4.7 Summary of the main results

To ease the reading of the experimental section, the main results and major takeaways are summarized here:

- the NTM achieves lower estimation error than the LSTM, while also using fewer learnable parameters (hence, a smaller memory footprint);
- a longer temporal context is beneficial for the NTM, whereas the prediction accuracy of the LSTM worsens as the sequences become longer;
- the piece-wise degradation function commonly used to label the CMAPSS dataset limits the predictions made by the model, which becomes unable to predict high RUL values at testing time;
- the NTM performs better than approaches following a similar architecture, especially when multiple fault conditions affect the measurements;
- the simple architecture proposed in this Chapter competes with state-of-the-art approaches which use additional optimization steps and more complex or deeper architectures;
- it takes longer to train the NTM, when compared to the LSTM;
- the NTM provides a prediction with a longer delay than the LSTM.

9.5 Conclusions and future work

In this Chapter, the Neural Turing Machines are thoroughly analyzed for the Remaining Useful Life estimation problem. The advantages provided by having access to the additional memory are confirmed by an extensive experimental section which shows that the

NTM can automatically extract useful features directly from the raw sensor measurements, obtaining better performance than MLP-, CNN-, and LSTM-based solutions: these performance are observed in terms of absolute estimation error, but also in three relative directions, that is parameter efficiency (fewer learnable parameters), memory efficiency (lesser memory footprint), and better usage of temporal context (longer sequences provide useful information to the NTM, whereas for the LSTM this does not hold). These results are empirically confirmed on two public datasets: the widely used C-MAPSS dataset [245] provided by NASA, and the recently released PHM Society 2020 Data Challenge [260]. The evidences also suggest that the NTM outperforms the LSTM when multiple operating conditions and fault modes affect the sensor measurements: this may be strategic in industrial settings, since complex machinery contain several deteriorating components. Moreover, since these faults may require more time to develop, being able to learn from a longer temporal context may be fundamental to catch them before they happen, therefore raising further interest towards the NTM. Furthermore, even though in this Chapter the NTM is used as the only feature extraction component, it can still achieve comparable and even competitive results to state-of-the-art techniques using ensemble of models, additional pretraining, *etc.* Therefore, combining the NTM with other state-of-the-art techniques may lead to additional improvements.

There is still room for research. In the experimental results, it is shown that the labeling function [99] commonly used for this task may not be the best choice and can highly affect the final performance. This is especially crucial for the datasets where the time series in the testing set have a higher groundtruth RUL than the maximum value used during the labeling step and therefore used to perform the training process. To address this, the community is actively seeking new solutions (*e.g.* Elsheikh et al. [63]) but, to date, no definitive solutions are found. Finally, some limitations of the NTM were also highlighted and contextualized to industrial scenarios, leaving further space for extensions and future works.

Bibliography

[1] Kfir Aberman, Mingyi Shi, Jing Liao, Dani Lischinski, Baoquan Chen, and Daniel Cohen-Or. Deep video-based performance cloning. In *Computer Graphics Forum*, volume 38, pages 219–233. Wiley Online Library, 2019.

[2] A. Agogino and K. Goebel. Milling data set, 2007. available at http://ti.arc.nasa.gov/project/prognostic-data-repository.

[3] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

[4] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6077–6086, 2018.

[5] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *IEEE/CVF International Conference on Computer Vision*, pages 2425–2433, 2015.

[6] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5297–5307, 2016.

[7] Joshua Attenberg, Panos Ipeirotis, and Foster Provost. Beat the machine: Challenging humans to find a predictive model's "unknown unknowns". *Journal of Data and Information Quality*, 6(1):1–17, 2015.

[8] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022.

[9] Giduthuri Sateesh Babu, Peilin Zhao, and Xiao-Li Li. Deep convolutional neural network based regression approach for estimation of remaining useful life. In *DASFAA*, pages 214–228. Springer, 2016.

[10] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York, 1999.

[11] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, 2015.

[12] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *IEEE/CVF International Conference on Computer Vision*, pages 1728–1738, 2021.

[13] Márcia Baptista, Helmut Prendinger, and Elsa Henriques. Prognostics in aeronautics with deep recurrent neural networks. In *PHM Society European Conference*, volume 5, 2020.

[14] Irwan Bello, William Fedus, Xianzhi Du, Ekin Dogus Cubuk, Aravind Srinivas, Tsung-Yi Lin, Jonathon Shlens, and Barret Zoph. Revisiting resnets: Improved training and scaling strategies. *Advances in Neural Information Processing Systems*, 34, 2021.

[15] Mathieu Blondel, Olivier Teboul, Quentin Berthet, and Josip Djolonga. Fast differentiable sorting and ranking. In *International Conference on Machine Learning*, pages 950–959. PMLR, 2020.

[16] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[17] Nathan Bolander, Hai Qiu, Neil Eklund, Ed Hindle, and Taylor Rosenfeld. Physics-based remaining useful life prediction for aircraft engine bearing prognosis. In *Annual Conference of the PHM Society*, volume 1, 2009.

[18] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 95–104, 2016.

[19] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[20] Christopher Burges, Robert Ragno, and Quoc Le. Learning to rank with nonsmooth cost functions. *Advances in Neural Information Processing Systems*, 19:193–200, 2006.

[21] Christopher Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *International Conference on Machine Learning*, pages 89–96, 2005.

[22] Carl S Byington, Susan E George, and G William Nickerson. Prognostic issues for rotorcraft health and usage monitoring systems. *A Critical Link: Diagnosis to Prognosis*, page 93, 1997.

[23] Santiago Castro, Ruoyao Wang, Pingxuan Huang, Ian Stewart, Oana Ignat, Nan Liu, Jonathan Stroud, and Rada Mihalcea. Fiber: Fill-in-the-blanks as a challenging video understanding evaluation framework. In *Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2925–2940, 2022.

[24] Nino Cauli and Diego Reforgiato Recupero. Survey on videos data augmentation for deep learning models. *Future Internet*, 14(3):93, 2022.

[25] L. Ceci. Hours of video uploaded to youtube every minute as of february 2020. `https://www.statista.com/statistics/259477/hours-of-video-uploaded-to-youtube-every-minute`, 2022. [Online; accessed 31-March-2022].

[26] Shizhe Chen, Yida Zhao, Qin Jin, and Qi Wu. Fine-grained video-text retrieval with hierarchical graph reasoning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10638–10647, June 2020.

[27] Shizhe Chen, Yida Zhao, Qin Jin, and Qi Wu. Fine-grained video-text retrieval with hierarchical graph reasoning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10638–10647, 2020.

[28] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pages 1597–1607. PMLR, 2020.

[29] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: a deep quadruplet network for person re-identification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 403–412, 2017.

[30] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016.

[31] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer, 2020.

[32] Zhen Chen, Yaping Li, Tangbin Xia, and Ershun Pan. Hidden markov model with auto-correlated observations for remaining useful life prediction and optimal maintenance policy. *Reliability Engineering & System Safety*, 184:123–136, 2019.

[33] De Cheng, Yihong Gong, Sanping Zhou, Jinjun Wang, and Nanning Zheng. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1335–1344, 2016.

[34] Tsz-Him Cheung and Dit-Yan Yeung. Modals: Modality-agnostic automated data augmentation in the latent space. In *International Conference on Learning Representations*, 2020.

[35] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734, 2014.

[36] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 539–546. IEEE, 2005.

[37] Alexandra Coppe, Matthew J Pais, Raphael T Haftka, and Nam H Kim. Using a simple crack growth model in predicting remaining useful life. *Journal of Aircraft*, 49(6):1965–1973, 2012.

[38] J. Cornelius, B. Brockner, S. H. Hong, Y. Wang, K. Pant, and J. Ball. Estimating and leveraging uncertainties in deep learning for remaining useful life prediction in mechanical systems. In *ICPHM*, pages 1–8, 2020.

[39] David Cossock and Tong Zhang. Statistical analysis of bayes optimal subset ranking. *IEEE Transactions on Information Theory*, 54(11):5140–5154, 2008.

[40] Ioana Croitoru, Simion-Vlad Bogolin, Marius Leordeanu, Hailin Jin, Andrew Zisserman, Samuel Albanie, and Yang Liu. Teachtext: Crossmodal generalized distillation for text-video retrieval. In *IEEE/CVF International Conference on Computer Vision*, pages 11583–11593, 2021.

[41] Yin Cui, Guandao Yang, Andreas Veit, Xun Huang, and Serge Belongie. Learning to evaluate image captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5804–5812, 2018.

[42] Marco Cuturi, Olivier Teboul, and Jean-Philippe Vert. Differentiable ranks and sorting using optimal transport. *Advances in Neural Information Processing Systems*, 2019.

[43] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893. Ieee, 2005.

[44] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[45] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Evangelos Kazakos, Jian Ma, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Rescaling egocentric vision. *International Journal of Computer Vision*, 2021.

[46] Dima Damen, Adriano Fragomeni, Jonathan Munro, Toby Perrett, Daniel Whettam, Michael Wray, Antonino Furnari, Giovanni Maria Farinella, and Davide Moltisanti. Epic-kitchens-100- 2021 challenges report. Technical report, University of Bristol, 2021.

[47] Dima Damen, Adriano Fragomeni, Toby Perrett, Daniel Whettam, Michael Wray, Bin Zhu, Antonino Furnari, Giovanni Maria Farinella, and Davide Moltisanti. Epic-kitchens-100- 2022 challenges report. Technical report, University of Bristol, 2022.

[48] Long Hoang Dang, Thao Minh Le, Vuong Le, and Truyen Tran. Object-centric representation learning for video question answering. In *International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2021.

[49] Eustasio Del Barrio, Juan A Cuesta-Albertos, and Carlos Matrán. An optimal transportation approach for assessing almost stochastic order. In *The Mathematics of the Uncertain*, pages 33–44. Springer, 2018.

[50] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. Ieee, 2009.

[51] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in Neural Information Processing Systems*, pages 1486–1494, 2015.

[52] Karan Desai and Justin Johnson. Virtex: Learning visual representations from textual annotations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11162–11173, 2021.

[53] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

[54] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

[55] Aiden R Doherty, Steve E Hodges, Abby C King, Alan F Smeaton, Emma Berry, Chris JA Moulin, Siân Lindley, Paul Kelly, and Charlie Foster. Wearable cameras in health: the state of the art and future possibilities. *American journal of preventive medicine*, 44(3), 2013.

[56] Jianfeng Dong, Xirong Li, Chaoxi Xu, Gang Yang, and Xun Wang. Feature relearning with data augmentation for content-based video recommendation. In *ACM International Conference on Multimedia*, pages 2058–2062, 2018.

[57] Jianfeng Dong, Xirong Li, Chaoxi Xu, Xun Yang, Gang Yang, Xun Wang, and Meng Wang. Dual encoding for video retrieval by text. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[58] Jianfeng Dong, Xun Wang, Leimin Zhang, Chaoxi Xu, Gang Yang, and Xirong Li. Feature re-learning with data augmentation for video relevance prediction. *IEEE Transactions on Knowledge and Data Engineering*, 33(5):1946–1959, 2019.

[59] Xinzhi Dong, Chengjiang Long, Wenju Xu, and Chunxia Xiao. Dual graph convolutional networks with transformer and curriculum learning for image captioning. In *ACM International Conference on Multimedia*, pages 2615–2624, 2021.

[60] Rotem Dror, Segev Shlomov, and Roi Reichart. Deep dominance - how to properly compare deep neural models. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2773–2785. Association for Computational Linguistics, 2019.

[61] Maksim Dzabraev, Maksim Kalashnikov, Stepan Komkov, and Aleksandr Petiushko. Mdmmt: Multidomain multimodal transformer for video retrieval. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (Workshop)*, pages 3354–3363, 2021.

[62] André Listou Ellefsen, Emil Bjørlykhaug, Vilmar Æsøy, Sergey Ushakov, and Houxiang Zhang. Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. *Reliab. Eng. Syst. Saf.*, 183:240–251, 2019.

[63] Ahmed Elsheikh, Soumaya Yacout, and Mohamed-Salah Ouali. Bidirectional handshaking lstm for remaining useful life prediction. *Neurocomputing*, 323:148–156, 2019.

[64] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021.

[65] Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *nature*, 542(7639):115–118, 2017.

[66] Alexander Richard Fabbri, Simeng Han, Haoyuan Li, Haoran Li, Marjan Ghazvininejad, Shafiq Joty, Dragomir Radev, and Yashar Mehdad. Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 704–717, 2021.

[67] A. Falcon, G. D'Agostino, G. Serra, G. Brajnik, and C. Tasso. A neural turing machine-based approach to remaining useful life estimation. In *ICPHM*, pages 1–8, 2020.

[68] Alex Falcon, Giovanni D'Agostino, Giuseppe Serra, Giorgio Brajnik, and Carlo Tasso. A dual-stream architecture based on neural turing machine and attention for the remaining useful life estimation problem. In *PHM Society European Conference*, volume 5, pages 10–10, 2020.

[69] Alex Falcon, Giuseppe Serra, Sergio Escalera, and Oswald Lanz. Uniud-fbk-ub-unibz submission to the epic-kitchens-100 multi-instance retrieval challenge 2022. *arXiv preprint arXiv:2206.10903*, 2022.

[70] Alex Falcon, Giuseppe Serra, and Oswald Lanz. Learning video retrieval models with relevance-aware online mining. In *International Conference on Image Analysis and Processing*, pages 182—-194, 2022.

[71] Alex Falcon, Swathikiran Sudhakaran, Giuseppe Serra, Sergio Escalera, and Oswald Lanz. Relevance-based margin for contrastively-trained video retrieval models. In *International Conference on Multimedia Retrieval*, pages 146—-157, 2022.

[72] Chenyou Fan. Egovqa-an egocentric video question answering benchmark dataset. In *IEEE/CVF International Conference on Computer Vision (Workshop)*, 2019.

[73] Chenyou Fan, Xiaofan Zhang, Shu Zhang, Wensheng Wang, Chi Zhang, and Heng Huang. Heterogeneous memory enhanced multimodal attention model for video question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[74] Zhiwei Fang, Jing Liu, Yong Li, Yanyuan Qiao, and Hanqing Lu. Improving visual question answering using dropout and enhanced question encoder. *Pattern Recognition*, 90:404–414, 2019.

[75] Zerun Feng, Zhimin Zeng, Caili Guo, and Zheng Li. Exploiting visual semantic reasoning for video-text retrieval. In *International Conference on International Joint Conferences on Artificial Intelligence*, pages 1005–1011, 2021.

[76] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *European Conference on Computer Vision*, pages 214–229. Springer, 2020.

[77] Rinon Gal, Or Patashnik, Haggai Maron, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *ACM Transactions on Graphics*, 41(4):1–13, 2022.

[78] Federico A Galatolo, Mario GCA Cimino, and Gigliola Vaglini. Generating images from caption and vice versa via clip-guided generative latent space search. In *International Conference on Image Processing and Vision Engineering*, volume 1, pages 166–174. SCITEPRESS-Science and Technology Publications, Lda, 2021.

[79] Jiyang Gao, Runzhou Ge, Kan Chen, and Ram Nevatia. Motion-appearance co-memory networks for video question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[80] Noa Garcia, Mayu Otani, Chenhui Chu, and Yuta Nakashima. Knowit vqa: Answering knowledge-based questions about videos. *National Conference of the American Association for Artificial Intelligence*, 34(07):10826–10834, 2020.

[81] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F Liu, Matthew E Peters, Michael Schmitz, and Luke Zettlemoyer. Allennlp: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software)*, pages 1–6, 2018.

[82] Yuying Ge, Yixiao Ge, Xihui Liu, Dian Li, Ying Shan, Xiaohu Qie, and Ping Luo. Bridging video-text retrieval with multiple choice questions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16167–16176, 2022.

[83] Donald Geman, Stuart Geman, Neil Hallonquist, and Laurent Younes. Visual turing test for computer vision systems. *National Academy of Sciences*, 112(12):3618–3623, 2015.

[84] Deepti Ghadiyaram, Du Tran, and Dhruv Mahajan. Large-scale weakly-supervised pre-training for video action recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12046–12055, 2019.

[85] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In Z Ghahramani, M Welling, C Cortes, N D Lawrence, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems*, pages 2672–2680. Curran Associates, Inc., 2014.

[86] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6904–6913, 2017.

[87] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[88] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[89] Frank L Greitzer, Lars J Kangas, Kristine M Terrones, Melody A Maynard, Bary W Wilson, Ronald A Pawlowski, Daniel R Sisk, and Newton B Brown. Gas turbine engine health monitoring and prognostics. In *International Society of Logistics (SOLE) 1999 Symposium, Las Vegas, Nevada*, volume 30, pages 1–7, 1999.

[90] Aditya Grover, Eric Wang, Aaron Zweig, and Stefano Ermon. Stochastic optimization of sorting networks via continuous relaxations. In *International Conference on Learning Representations*, 2018.

[91] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image

synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10696–10706, 2022.

[92] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *International Conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

[93] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *IEEE Computer Society Computer Vision and Pattern Recognition*, volume 2, pages 1735–1742. IEEE, 2006.

[94] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018.

[95] Ben Harwood, Vijay Kumar BG, Gustavo Carneiro, Ian Reid, and Tom Drummond. Smart mining for deep metric learning. In *IEEE/CVF International Conference on Computer Vision*, pages 2821–2829, 2017.

[96] Feng He, Qi Wang, Zhifan Feng, Wenbin Jiang, Yajuan Lü, Yong Zhu, and Xiao Tan. Improving video retrieval by adaptive margin. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1359–1368, 2021.

[97] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[98] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[99] F. O. Heimes. Recurrent neural networks for remaining useful life estimation. In *ICPHM*, 2008.

[100] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[101] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium – supplementary material. *Advances in Neural Information Processing Systems*, 30, 2017.

[102] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[103] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[104] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997.

[105] Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee. Inferring semantic layout for hierarchical text-to-image synthesis. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7986–7994, 2018.

[106] Chiori Hori, Takaaki Hori, Teng-Yok Lee, Ziming Zhang, Bret Harsham, John R Hershey, Tim K Marks, and Kazuhiko Sumi. Attention-based multimodal fusion for video description. In *IEEE/CVF International Conference on Computer Vision*, 2017.

[107] MD Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CSUR)*, 51(6):118, 2019.

[108] Mengru Hou, Dechang Pi, and Bingrong Li. Similarity-based deep learning approach for remaining useful life prediction. *Measurement*, 159:107788, 2020.

[109] Sixing Hu, Mengdan Feng, Rang MH Nguyen, and Gim Hee Lee. Cvm-net: Cross-view matching network for image-based ground-to-aerial geo-localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7258–7267, 2018.

[110] Yuan-Ting Hu, Jiahong Wang, Raymond A Yeh, and Alexander G Schwing. Sailvos 3d: A synthetic dataset and baselines for object detection and 3d mesh reconstruction from video data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1418–1428, 2021.

[111] Deng Huang, Peihao Chen, Runhao Zeng, Qing Du, Mingkui Tan, and Chuang Gan. Location-aware graph convolutional networks for video question answering. In *National Conference of the American Association for Artificial Intelligence*, pages 11021–11028, 2020.

[112] Hong-Zhong Huang, Hai-Kun Wang, Yan-Feng Li, Longlong Zhang, and Zhiliang Liu. Support vector machine based estimation of remaining useful life: current research status and future trends. *Journal of Mechanical Science and Technology*, 29(1):151–163, 2015.

[113] Hochul Hwang, Cheongjae Jang, Geonwoo Park, Junghyun Cho, and Ig-Jae Kim. Eldersim: A synthetic data generation platform for human action recognition in eldercare applications. *IEEE Access*, 2021.

[114] Boulent M Imam and Marios K Chryssanthopoulos. Causes and consequences of metallic bridge failures. *Structural engineering international*, 22(1):93–98, 2012.

[115] Kürşat Ince, Engin Sirkeci, and Yakup Genç. Remaining useful life prediction for experimental filtration system: A data challenge. In *PHM Society European Conference*, volume 5, 2020.

[116] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456. PMLR, 2015.

[117] Takashi Isobe, Jian Han, Fang Zhuz, Yali Liy, and Shengjin Wang. Intra-clip aggregation for video person re-identification. In *IEEE International Conference on Image Processing*, pages 2336–2340. IEEE, 2020.

[118] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5967–5976, 2017.

[119] Nikita Jaipuria, Xianling Zhang, Rohan Bhasin, Mayar Arafa, Punarjay Chakravarty, Shubham Shrivastava, Sagar Manglani, and Vidya N Murali. Deflating dataset bias using synthetic data augmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 772–773, 2020.

[120] Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[121] Andrew KS Jardine, Daming Lin, and Dragan Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *MSSP*, 20(7):1483–1510, 2006.

[122] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

[123] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

[124] Pin Jiang and Yahong Han. Reasoning with heterogeneous graph alignment for video question answering. In *National Conference of the American Association for Artificial Intelligence*, pages 11109–11116, 2020.

[125] Qing-Yuan Jiang, Yi He, Gen Li, Jian Lin, Lei Li, and Wu-Jun Li. Svd: A large-scale short video dataset for near-duplicate video retrieval. In *IEEE/CVF International Conference on Computer Vision*, pages 5281–5289, 2019.

[126] Yuming Jiang, Shuai Yang, Haonan Qju, Wayne Wu, Chen Change Loy, and Ziwei Liu. Text2human: Text-driven controllable human image generation. *ACM Transactions on Graphics*, 41(4):1–11, 2022.

[127] Weike Jin, Zhou Zhao, Pengcheng Zhang, Jieming Zhu, Xiuqiang He, and Yueting Zhuang. Hierarchical cross-modal graph consistency learning for video-text retrieval. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1114–1124, 2021.

[128] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[129] Kushal Kafle and Christopher Kanan. Visual question answering: Datasets, algorithms, and future challenges. *Computer Vision and Image Understanding*, 163:3–20, 2017.

[130] Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan. Data augmentation for visual question answering. In *International Conference on Natural Language Generation*, pages 198–202, 2017.

[131] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3128–3137, 2015.

[132] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2014.

[133] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[134] Evangelos Kazakos, Arsha Nagrani, Andrew Zisserman, and Dima Damen. Epic-fusion: Audio-visual temporal binding for egocentric action recognition. In *IEEE/CVF International Conference on Computer Vision*, pages 5492–5501, 2019.

[135] Hyounghun Kim, Zineng Tang, and Mohit Bansal. Dense-caption matching and frame-selection gating for temporal localization in videoqa. In *Annual Meeting of the Association for Computational Linguistics*, pages 4812–4822, 2020.

[136] Junyeong Kim, Minuk Ma, Kyungsu Kim, Sungjin Kim, and Chang D Yoo. Gaining extra supervision via multi-task learning for multi-modal video question answering. In *International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2019.

[137] Junyeong Kim, Minuk Ma, Trung Pham, Kyungsu Kim, and Chang D Yoo. Modality shifting attention network for multi-modal video question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10106–10115, 2020.

[138] Kyung-Min Kim, Seong-Ho Choi, Jin-Hwa Kim, and Byoung-Tak Zhang. Multimodal dual attention memory for video story question answering. In *European Conference on Computer Vision*, 2018.

[139] Kyung-Min Kim, Min-Oh Heo, Seong-Ho Choi, and Byoung-Tak Zhang. Deepstory: video story qa by deep embedded memory networks. In *International Joint Conferences on Artificial Intelligence*, 2017.

[140] Nayoung Kim, Seong Jong Ha, and Je-Won Kang. Video question answering using language-guided deep compressed-domain video feature. In *IEEE/CVF International Conference on Computer Vision*, pages 1708–1717, 2021.

[141] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[142] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*, 2017.

[143] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in Neural Information Processing Systems*, pages 3294–3302, 2015.

[144] Jing Yu Koh, Jason Baldridge, Honglak Lee, and Yinfei Yang. Text-to-image generation grounded by fine-grained user attention. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 237–246, 2021.

[145] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.

[146] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[147] Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and Wlliam Campbell. A closer look at feature space data augmentation for few-shot intent classification. In *Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 1–10, 2019.

[148] Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, 2019.

[149] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *IEEE*, 86(11):2278–2324, 1998.

[150] Seungmin Lee, Dongwan Kim, and Bohyung Han. Cosmo: Content-style modulation for image retrieval with text feedback. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 802–812, 2021.

[151] Jie Lei, Linjie Li, Luowei Zhou, Zhe Gan, Tamara L Berg, Mohit Bansal, and Jingjing Liu. Less is more: Clipbert for video-and-language learning via sparse sampling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7331–7341, 2021.

[152] Jie Lei, Liwei Wang, Yelong Shen, Dong Yu, Tamara Berg, and Mohit Bansal. Mart: Memory-augmented recurrent transformer for coherent video paragraph captioning. In *Annual Meeting of the Association for Computational Linguistics*, pages 2603–2614, 2020.

[153] Jie Lei, Licheng Yu, Mohit Bansal, and Tamara Berg. Tvqa: Localized, compositional video question answering. In *Conference on Empirical Methods in Natural Language Processing*, pages 1369–1379, 2018.

[154] Jie Lei, Licheng Yu, Tamara L Berg, and Mohit Bansal. Tvqa+: Spatio-temporal grounding for video question answering. In *Annual Meeting of the Association for Computational Linguistics*, pages 8211–8225, 2019.

[155] Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *annual International Conference on Systems documentation*, pages 24–26, 1986.

[156] Jie Li, Mingqiang Yang, Yupeng Liu, Yanyan Wang, Qinghe Zheng, and Deqiang Wang. Dynamic hand gesture recognition using multi-direction 3d convolutional neural networks. *Engineering Letters*, 27(3), 2019.

[157] Linjie Li, Yen-Chun Chen, Yu Cheng, Zhe Gan, Licheng Yu, and Jingjing Liu. Hero: Hierarchical encoder for video+ language omni-representation pre-training. In *Conference on Empirical Methods in Natural Language Processing*, pages 2046–2065, 2020.

[158] Linjie Li, Jie Lei, Zhe Gan, Licheng Yu, Yen-Chun Chen, Rohit Pillai, Yu Cheng, Luowei Zhou, Xin Eric Wang, William Yang Wang, et al. Value: A multi-task benchmark for video-and-language understanding evaluation. In *Advances in Neural Information Processing Systems - Track on Datasets and Benchmarks*, 2021.

[159] Mingming Li, Shuai Zhang, Fuqing Zhu, Wanhui Qian, Liangjun Zang, Jizhong Han, and Songlin Hu. Symmetric metric learning with adaptive margin for recommendation. *National Conference of the American Association for Artificial Intelligence*, 34(04):4634–4641, 2020.

[160] Naipeng Li, Yaguo Lei, Tao Yan, Ningbo Li, and Tianyu Han. A wiener-process-model-based method for remaining useful life prediction considering unit-to-unit variability. *IEEE Transactions on Industrial Electronics*, 66(3):2092–2101, 2018.

[161] Wenbo Li, Pengchuan Zhang, Lei Zhang, Qiuyuan Huang, Xiaodong He, Siwei Lyu, and Jianfeng Gao. Object-driven text-to-image synthesis via adversarial training. *CoRR*, abs/1902.10740, 2019.

[162] Xiang Li, Qian Ding, and Jian-Qiao Sun. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliab. Eng. Syst. Saf.*, 172:1–11, 2018.

[163] Xiangpeng Li, Jingkuan Song, Lianli Gao, Xianglong Liu, Wenbing Huang, Xiangnan He, and Chuang Gan. Beyond rnns: Positional self-attention with co-attention for video question answering. *National Conference of the American Association for Artificial Intelligence*, 33(01):8658–8665, 2019.

[164] Xirong Li, Fangming Zhou, Chaoxi Xu, Jiaqi Ji, and Gang Yang. Sea: Sentence encoder assembly for video retrieval by textual queries. *IEEE Transactions on Multimedia*, 23:4351–4362, 2020.

[165] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer, 2020.

[166] Yuncheng Li, Yale Song, Liangliang Cao, Joel Tetreault, Larry Goldberg, Alejandro Jaimes, and Jiebo Luo. Tgif: A new dataset and benchmark on animated gif description. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4641–4650, 2016.

[167] Zhang Li, Sung Flood, Liu Feng, Xiang Tao, Gong Shaogang, Yang Yongxin, and Hospedales Timothy M. Actor-critic sequence training for image captioning. In *Advances in Neural Information Processing Systems (Workshop)*, 2017.

[168] Wentong Liao, Kai Hu, Michael Ying Yang, and Bodo Rosenhahn. Text to image generation with semantic-spatial aware gan. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18187–18196, 2022.

[169] Kevin Qinghong Lin, Alex Jinpeng Wang, Rui Yan, Eric Zhongcong Xu, Rongcheng Tu, Yanru Zhu, Wenzhe Zhao, Weijie Kong, Chengfei Cai, Hongfa Wang, et al. Egocentric video-language pretraining@ epic-kitchens-100 multi-instance retrieval challenge 2022. *arXiv preprint arXiv:2207.01334*, 2022.

[170] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *European Conference on Computer Vision*, pages 740–755, 2014.

[171] Fei Liu, Jing Liu, Weining Wang, and Hanqing Lu. Hair: Hierarchical visual-semantic relational reasoning for video question answering. In *IEEE/CVF International Conference on Computer Vision*, pages 1698–1707, 2021.

[172] Lijun Liu, Lan Wang, and Zhen Yu. Remaining useful life estimation of aircraft engines based on deep convolution neural network and lightgbm combination model. *International Journal of Computational Intelligence Systems*, 14(1):1–10, 2021.

[173] Song Liu, Haoqi Fan, Shengsheng Qian, Yiru Chen, Wenkui Ding, and Zhongyuan Wang. Hit: Hierarchical transformer with momentum contrast for video-text retrieval. *IEEE/CVF International Conference on Computer Vision*, 2021.

[174] Xiaofeng Liu, Yang Zou, Lingsheng Kong, Zhihui Diao, Junliang Yan, Jun Wang, Site Li, Ping Jia, and Jane You. Data augmentation via latent space interpolation for image classification. In *International Conference on Pattern Recognition*, pages 728–733. IEEE, 2018.

[175] Yang Liu, Samuel Albanie, Arsha Nagrani, and Andrew Zisserman. Use what you have: Video retrieval using representations from collaborative experts. *British Machine Vision Conference*, 2019.

[176] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv:1907.11692*, 2019.

[177] Roman Łomowski and Szczepan Hummel. A method to estimate the remaining useful life of a filter using a hybrid approach based on kernel regression and simple statistics. In *PHM Society European Conference*, volume 5, 2020.

[178] Shayne Longpre, Yu Wang, and Chris DuBois. How effective is task-agnostic data augmentation for pretrained transformers? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4401–4411, 2020.

[179] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[180] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-task vision and language representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10437–10446, 2020.

[181] Qinyuan Luo, Yuanhong Chang, Jinglong Chen, Hongjie Jing, Haixin Lv, and Tongyang Pan. Multiple degradation mode analysis via gated recurrent unit mode recognizer and life predictors for complex equipment. *Computers in Industry*, 123:103332, 2020.

[182] Ying Luo, Hai Zhao, and Junlang Zhan. Named entity recognition only from word embeddings. *Conference on Empirical Methods in Natural Language Processing*, 2020.

[183] Jianhua Lyu, Rongrong Ying, Ningyun Lu, and Baili Zhang. Remaining useful life estimation with multiple local similarities. *Eng. Appl. Artif. Intell.*, 95:103849, 2020.

[184] Yiwei Ma, Guohai Xu, Xiaoshuai Sun, Ming Yan, Ji Zhang, and Rongrong Ji. X-clip: End-to-end multi-grained contrastive learning for video-text retrieval. In *ACM International Conference on Multimedia*, page 638–647, New York, NY, USA, 2022. Association for Computing Machinery.

[185] James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1. Oakland, CA, USA, 1967.

[186] Tegan Maharaj, Nicolas Ballas, Anna Rohrbach, Aaron Courville, and Christopher Pal. A dataset and exploration of models for understanding video data through fill-in-the-blank question-answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6884–6893, 2017.

[187] Aroma Mahendru, Viraj Prabhu, Akrit Mohapatra, Dhruv Batra, and Stefan Lee. The promise of premise: Harnessing question premises in visual question answering. In *Conference on Empirical Methods in Natural Language Processing*, pages 926–935, 2017.

[188] Elman Mansimov, Emilio Parisotto, Lei Jimmy Ba, and Ruslan Salakhutdinov. Generating images from captions with attention. In *International Conference on Learning Representations*, 2016.

[189] Jianguo Mao, Wenbin Jiang, Xiangdong Wang, Zhifan Feng, Yajuan Lyu, Hong Liu, and Yong Zhu. Dynamic multistep reasoning based on video scene graph for

video question answering. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3894–3904, 2022.

[190] Bernard Marr. How much data do we create every day? the mind-blowing stats everyone should read. `https://bernardmarr.com/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-shou` 2022. [Online; accessed 30-November-2022].

[191] Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, 2019.

[192] Kamal Medjaher, Diego Alejandro Tobon-Mejia, and Noureddine Zerhouni. Remaining useful life estimation of critical components with application to bearings. *IEEE Transactions on Reliability*, 61(2):292–302, 2012.

[193] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9879–9889, 2020.

[194] Antoine Miech, Ivan Laptev, and Josef Sivic. Learning a text-video embedding from incomplete and heterogeneous data. *arXiv preprint arXiv:1804.02516*, 2018.

[195] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *IEEE/CVF International Conference on Computer Vision*, pages 2630–2640, 2019.

[196] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, 2013.

[197] P Milillo, G Giardina, D Perissin, G Milillo, A Coletta, and C. Terranova. Pre-collapse space geodetic observations of critical infrastructure: The morandi bridge, genoa, italy. *Remote Sensing*, 11:1403, 2019.

[198] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[199] Junghyun Min, R Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. Syntactic data augmentation increases robustness to inference heuristics. In *Annual Meeting of the Association for Computational Linguistics*, pages 2339–2352, 2020.

[200] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[201] Niluthpol Chowdhury Mithun, Juncheng Li, Florian Metze, and Amit K Roy-Chowdhury. Learning joint embedding with multimodal cues for cross-modal video-text retrieval. In *ACM International Conference on Multimedia Retrieval*, pages 19–27, 2018.

[202] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter Battaglia. Generating images with sparse representations. In *International Conference on Machine Learning*, pages 7958–7968. PMLR, 2021.

[203] Anh Nguyen, Jeff Clune, Yoshua Bengio, Alexey Dosovitskiy, and Jason Yosinski. Plug & play generative networks: Conditional iterative generation of images in latent space. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3510–3520, 2017.

[204] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org, 2017.

[205] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[206] Celestino Ordóñez, Fernando Sánchez Lasheras, Javier Roca-Pardiñas, and Francisco Javier de Cos Juez. A hybrid arima–svm model for the study of the remaining useful life of aircraft engines. *Journal of Computational and Applied Mathematics*, 346:184–191, 2019.

[207] Tian Pan, Yibing Song, Tianyu Yang, Wenhao Jiang, and Wei Liu. Videomoco: Contrastive video representation learning with temporally adversarial examples. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11205–11214, 2021.

[208] Yubin Pan, Rongjing Hong, Jie Chen, and Weiwei Wu. A hybrid dbn-som-pf-based prognostic approach of remaining useful life for wind turbine gearbox. *Renewable Energy*, 152:138–154, 2020.

[209] Shantipriya Parida and Petr Motlicek. Abstract text summarization: A low resource challenge. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 5994–5998, 2019.

[210] Jungin Park, Jiyoung Lee, and Kwanghoon Sohn. Bridge to answer: Structure-aware graph interaction network for video question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15526–15535, 2021.

[211] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *IEEE/CVF International Conference on Computer Vision*, pages 2085–2094, 2021.

[212] Mandela Patrick, Po-Yao Huang, Yuki Asano, Florian Metze, Alexander Hauptmann, Joao Henriques, and Andrea Vedaldi. Support-set bottlenecks for video-text representation learning. *International Conference on Learning Representations*, 2020.

[213] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[214] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

[215] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.

[216] Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. Mad-x: An adapter-based framework for multi-task cross-lingual transfer. In *Conference on Empirical Methods in Natural Language Processing*, pages 7654–7673, 2020.

[217] Hieu Pham, Xinyi Wang, Yiming Yang, and Graham Neubig. Meta back-translation. In *International Conference on Learning Representations*, 2020.

[218] AJ Piergiovanni, Kairo Morton, Weicheng Kuo, Michael S Ryoo, and Anelia Angelova. Video question answering with iterative video-text co-tokenization. In *European Conference on Computer Vision*, pages 76–94. Springer, 2022.

[219] L Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: Unbiased boosting with categorical features. arxiv 2017. *arXiv preprint arXiv:1706.09516*, 2017.

[220] Rui Qian, Tianjian Meng, Boqing Gong, Ming-Hsuan Yang, Huisheng Wang, Serge Belongie, and Yin Cui. Spatiotemporal contrastive video representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6964–6974, 2021.

[221] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. Mirrorgan: Learning text-to-image generation by redescription. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1505–1514, 2019.

[222] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. Mirrorgan: Learning text-to-image generation by redescription. *CoRR*, abs/1903.05854, 2019.

[223] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[224] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[225] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[226] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*, 2016.

[227] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in Neural Information Processing Systems*, 32, 2019.

[228] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[229] Scott E. Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. Learning deep representations of fine-grained visual descriptions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 49–58, 2016.

[230] Scott E. Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, and Honglak Lee. Learning what and where to draw. In *Advances in Neural Information Processing Systems*, pages 217–225, 2016.

[231] Scott E. Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International Conference on Machine Learning*, pages 1060–1069, 2016.

[232] Scott E. Reed, Aaron van den Oord, Nal Kalchbrenner, Victor Bapst, Matt M. Botvinick, and Nando de Freitas. Generating Interpretable Images with Controllable Structure, 2017.

[233] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 28, 2015.

[234] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1179–1195, 2017.

[235] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[236] Karsten Roth, Oriol Vinyals, and Zeynep Akata. Integrating language guidance into vision-based deep metric learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16177–16189, 2022.

[237] Shulan Ruan, Yong Zhang, Kun Zhang, Yanbo Fan, Fan Tang, Qi Liu, and Enhong Chen. Dae-gan: Dynamic aspect-aware gan for text-to-image synthesis. In *IEEE/CVF International Conference on Computer Vision*, pages 13960–13969, 2021.

[238] Arka Sadhu, Kan Chen, and Ram Nevatia. Video question answering with phrases via semantic roles. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2460–2478, 2021.

[239] Dimitrios Sakkos, Hubert PH Shum, and Edmond SL Ho. Illumination-based data augmentation for robust background subtraction. In *International Conference on Software, Knowledge, Information Management and Applications*, pages 1–8. IEEE, 2019.

[240] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2226–2234, 2016.

[241] Arthur L Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3:210–229, 1959.

[242] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *Advances in Neural Information Processing Systems (Workshop)*, 2019.

[243] Mert Bulent Sariyildiz, Julien Perez, and Diane Larlus. Learning visual representations with caption annotations. In *European Conference on Computer Vision*, pages 153–170. Springer, 2020.

[244] Burak Satar, Hongyuan Zhu, Hanwang Zhang, and Joo Hwee Lim. Exploiting semantic role contextualized video features for multi-instance text-video retrieval epic-kitchens-100 multi-instance retrieval challenge 2022. *arXiv preprint arXiv:2206.14381*, 2022.

[245] A. Saxena and K. Goebel. Turbofan engine degradation simulation data set. *NASA Ames Prognostics Data Repository*, 2008.

[246] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, pages 1–9. IEEE, 2008.

[247] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE/CVF Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[248] David Semedo and João Magalhães. Cross-modal subspace learning with scheduled adaptive margin constraints. In *ACM International Conference on Multimedia*, pages 75–83, 2019.

[249] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, 2016.

[250] Ahjeong Seo, Gi-Cheon Kang, Joonhan Park, and Byoung-Tak Zhang. Attend what you need: Motion-appearance synergistic networks for video question answering. In *Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6167–6177, 2021.

[251] Miran Seok, Hye-Jeong Song, Chan-Young Park, Jong-Dae Kim, and Yu-seop Kim. Named entity recognition using word embedding as a feature. *International Journal of Software Engineering and Its Applications*, 10(2):93–104, 2016.

[252] Meet Shah, Xinlei Chen, Marcus Rohrbach, and Devi Parikh. Cycle-consistency for robust visual question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6649–6658, 2019.

[253] Rakshith Shetty, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, and Bernt Schiele. Speaking the same language: Matching machine to human captions by adversarial training. In *IEEE/CVF International Conference on Computer Vision*, pages 4155–4164, 2017.

[254] Zhan Shi, Hui Liu, and Xiaodan Zhu. Enhancing descriptive image captioning with natural language inference. In *Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 269–277, 2021.

[255] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

[256] Connor Shorten, Taghi M Khoshgoftaar, and Borko Furht. Text data augmentation for deep learning. *Journal of big Data*, 8(1):1–34, 2021.

[257] Nina Shvetsova, Brian Chen, Andrew Rouditchenko, Samuel Thomas, Brian Kingsbury, Rogerio S Feris, David Harwath, James Glass, and Hilde Kuehne. Everything at once-multi-modal fusion transformer for video retrieval. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20020–20029, 2022.

[258] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[259] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[260] PHM Society. Phm society 2020 data challenge, 2020. https://phm-europe.org/data-challenge-2020 (last accessed: 23 April 2022).

[261] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[262] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, pages 1857–1865, 2016.

[263] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[264] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

[265] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv:1212.0402*, 2012.

[266] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.

[267] Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International Conference on Machine Learning*, pages 9120–9132. PMLR, 2020.

[268] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy, July 2019. Association for Computational Linguistics.

[269] Yumin Suh, Bohyung Han, Wonsik Kim, and Kyoung Mu Lee. Stochastic class-based hard example mining for deep metric learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7251–7259, 2019.

[270] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *IEEE/CVF International Conference on Computer Vision*, pages 7464–7473, 2019.

[271] Guanglu Sun, Lili Liang, Tianlin Li, Bo Yu, Meng Wu, and Bolun Zhang. Video question answering: a survey of models and datasets. *Mobile Networks and Applications*, pages 1–34, 2021.

[272] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[273] Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Movieqa: Understanding stories in movies through question-answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4631–4640, 2016.

[274] Wei Teng, Chen Han, Yankang Hu, Xin Cheng, Lei Song, and Yibing Liu. A robust model-based approach for bearing remaining useful life prognosis in wind turbines. *IEEE Access*, 8:47133–47143, 2020.

[275] Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. Winoground: Probing vision and language models for visio-linguistic compositionality. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5238–5248, 2022.

[276] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *IEEE/CVF International Conference on Computer Vision*, pages 4489–4497, 2015.

[277] Dennis Ulmer, Christian Hardmeier, and Jes Frellsen. deep-significance: Easy and meaningful signifcance testing in the age of neural networks. In *International Conference on Learning Representations (Workshop)*, 2022.

[278] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[279] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[280] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2015.

[281] Cheng Wang, Haojin Yang, Christian Bartz, and Christoph Meinel. Image captioning with deep bidirectional lstms. In *ACM International Conference on Multimedia*, pages 988–997, 2016.

[282] Jianyu Wang, Bingkun Bao, and Changsheng Xu. Dualvgr: A dual-visual graph reasoning unit for video question answering. *IEEE Transactions on Multimedia*, 2021.

[283] Jinjiang Wang, Jianxing Yan, Chen Li, Robert X Gao, and Rui Zhao. Deep heterogeneous gru model for predictive analytics in smart manufacturing: Application to tool wear prediction. *Computers in Industry*, 111:1–14, 2019.

[284] Jiujian Wang, Guilin Wen, Shaopu Yang, and Yongqiang Liu. Remaining useful life estimation in prognostics using deep bidirectional lstm neural network. In *PHM-Chongqing*, pages 1037–1042. IEEE, 2018.

[285] Liangliang Wang, Lianzheng Ge, Ruifeng Li, and Yajun Fang. Three-stream cnns for action recognition. *Pattern Recognition Letters*, 92:33–40, 2017.

[286] Shuo Wang, Dan Guo, Xin Xu, Li Zhuo, and Meng Wang. Cross-modality retrieval by joint correlation learning. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 15(2s):1–16, 2019.

[287] William Yang Wang and Diyi Yang. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *conference on empirical methods in natural language processing*, pages 2557–2563, 2015.

[288] Xiaohan Wang, Linchao Zhu, and Yi Yang. T2vlad: global-local sequence alignment for text-video retrieval. In *IEEE/CVF Computer Vision and Pattern Recognition*, pages 5079–5088, 2021.

[289] Xin Wang, Jiawei Wu, Junkun Chen, Lei Li, Yuan-Fang Wang, and William Yang Wang. Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. In *IEEE/CVF International Conference on Computer Vision*, pages 4581–4591, 2019.

[290] Yiren Wang, ChengXiang Zhai, and Hany Hassan Awadalla. Multi-task learning for multilingual neural machine translation. In *Conference on Empirical Methods in Natural Language Processing*, pages 1022–1034, 2020.

[291] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. In *International Conference on Learning Representations*, 2021.

[292] Zixu Wang, Yishu Miao, and Lucia Specia12. Cross-modal generative augmentation for visual question answering. In *British Machine Vision Conference*, 2021.

[293] Dongxu Wei, Xiaowei Xu, Haibin Shen, and Kejie Huang. Gac-gan: A general method for appearance-controllable human video motion transfer. *IEEE Transactions on Multimedia*, 23:2457–2470, 2020.

[294] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 6382–6388, 2019.

[295] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 5 1992.

[296] Thomas Winterbottom, Sarah Xiao, Alistair McLean, and Noura Al Moubayed. On modality bias in the tvqa dataset. *British Machine Vision Conference*, 2020.

[297] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. *arXiv:1910.03771*, 2019.

[298] Michael Wray, Hazel Doughty, and Dima Damen. On semantic similarity in video retrieval. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3650–3660, 2021.

[299] Michael Wray, Diane Larlus, Gabriela Csurka, and Dima Damen. Fine-grained action retrieval through multiple parts-of-speech embeddings. In *IEEE/CVF International Conference on Computer Vision*, pages 450–459, 2019.

[300] Chenfei Wu, Jian Liang, Lei Ji, Fan Yang, Yuejian Fang, Daxin Jiang, and Nan Duan. Nüwa: Visual synthesis pre-training for neural visual world creation. In *European Conference on Computer Vision*, 2021.

[301] Peng Wu, Xiangteng He, Mingqian Tang, Yiliang Lv, and Jing Liu. Hanet: Hierarchical alignment networks for video-text retrieval. In *ACM International Conference on Multimedia*, pages 3518–3527, 2021.

[302] Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer, 2019.

[303] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[304] Jun Xia, Yunwen Feng, Cheng Lu, Chengwei Fei, and Xiaofeng Xue. Lstm-based multi-layer self-attention method for remaining useful life estimation of mechanical systems. *Engineering Failure Analysis*, 125:105385, 2021.

[305] Tangbin Xia, Ya Song, Yu Zheng, Ershun Pan, and Lifeng Xi. An ensemble framework based on convolutional bi-directional lstm with multiple time windows for remaining useful life estimation. *Computers in Industry*, 115:103182, 2020.

[306] Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9777–9786, 2021.

[307] Ziang Xie, Sida I Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y Ng. Data noising as smoothing in neural network language models. In *International Conference on Learning Representations*, 2016.

[308] Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. Video question answering via gradually refined attention over appearance and motion. In *ACM International Conferece on Multimedia*, 2017.

[309] Hu Xu, Gargi Ghosh, Po-Yao Huang, Prahal Arora, Masoumeh Aminzadeh, Christoph Feichtenhofer, Florian Metze, and Luke Zettlemoyer. Vlm: Task-agnostic video-language model pre-training for video understanding. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4227–4239, 2021.

[310] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *IEEE/CVF Computer Vision and Pattern Recognition*, pages 5288–5296, 2016.

[311] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1316–1324, 2018.

[312] Zhenqi Xu, Jiani Hu, and Weihong Deng. Recurrent convolutional neural network for video classification. In *IEEE International Conference on Multimedia and Expo*, pages 1–6. IEEE, 2016.

[313] Hong Xuan, Abby Stylianou, Xiaotong Liu, and Robert Pless. Hard negative examples are hard, but useful. In *European Conference on Computer Vision*, pages 126–142, 2020.

[314] Hong Xuan, Abby Stylianou, and Robert Pless. Improved embeddings with easy positive triplet mining. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2474–2482, 2020.

[315] Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Just ask: Learning to answer questions from millions of narrated videos. In *IEEE/CVF International Conference on Computer Vision*, pages 1686–1697, 2021.

[316] Pinci Yang, Xin Wang, Xuguang Duan, Hong Chen, Runze Hou, Cong Jin, and Wenwu Zhu. Avqa: A dataset for audio-visual question answering on videos. In *ACM International Conference on Multimedia*, pages 3480–3491, 2022.

[317] Zekun Yang, Noa Garcia, Chenhui Chu, Mayu Otani, Yuta Nakashima, and Haruo Takemura. Bert representations for video question answering. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020.

[318] Hui Ye, Xiulong Yang, Martin Takác, Rajshekhar Sunderraman, and Shihao Ji. Improving text-to-image synthesis using contrastive learning. In *British Machine Vision Conference*, 2021.

[319] Guojun Yin, Bin Liu, Lu Sheng, Nenghai Yu, Xiaogang Wang, and Jing Shao. Semantics disentangling for text-to-image generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2327–2336, 2019.

[320] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. {LSUN:} Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *CoRR*, abs/1506.0, 2015.

[321] Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *National Conference of the American Association for Artificial Intelligence*, volume 33, pages 9127–9134, 2019.

[322] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.

[323] Heeseung Yun, Youngjae Yu, Wonsuk Yang, Kangil Lee, and Gunhee Kim. Pano-avqa: Grounded audio-visual question answering on 360° videos. In *IEEE/CVF International Conference on Computer Vision*, pages 2011–2021. IEEE, 2021.

[324] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *IEEE/CVF International Conference on computer vision*, pages 6023–6032, 2019.

[325] Sangdoo Yun, Seong Joon Oh, Byeongho Heo, Dongyoon Han, and Jinhyung Kim. Videomix: Rethinking data augmentation for video classification. *arXiv preprint arXiv:2012.03457*, 2020.

[326] Amir R Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J Guibas. Robust learning through cross-task consistency. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11197–11206, 2020.

[327] Kuo-Hao Zeng, Tseng-Hung Chen, Ching-Yao Chuang, Yuan-Hong Liao, Juan Carlos Niebles, and Min Sun. Leveraging video descriptions to learn video question answering. In *National Conference of the American Association for Artificial Intelligence*, 2017.

[328] Xunlin Zhan, Yangxin Wu, Xiao Dong, Yunchao Wei, Minlong Lu, Yichi Zhang, Hang Xu, and Xiaodan Liang. Product1m: Towards weakly supervised instance-level product retrieval via cross-modal pretraining. In *IEEE/CVF International Conference on Computer Vision*, pages 11782–11791, 2021.

[329] Bin Zhang, Shaohui Zhang, and Weihua Li. Bearing performance degradation assessment using long short-term memory recurrent network. *Computers in Industry*, 106:14–29, 2019.

[330] Han Zhang, Jing Yu Koh, Jason Baldridge, Honglak Lee, and Yinfei Yang. Cross-modal contrastive learning for text-to-image generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 833–842, 2021.

[331] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks. In *IEEE/CVF International Conference on Computer Vision*, 2017.

[332] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[333] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[334] Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Yin and yang: Balancing and answering binary visual questions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5014–5022, 2016.

[335] Qi Zhang, Zhen Lei, Zhaoxiang Zhang, and Stan Z Li. Context-aware attention network for image-text retrieval. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3536–3545, 2020.

[336] Yingying Zhang, Qiaoyong Zhong, Liang Ma, Di Xie, and Shiliang Pu. Learning incremental triplet margin for person re-identification. *National Conference of the American Association for Artificial Intelligence*, 33(01):9243–9250, 2019.

[337] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[338] Zizhao Zhang, Yuanpu Xie, and Lin Yang. Photographic text-to-image synthesis with a hierarchically-nested adversarial network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6199–6208, 2018.

[339] Zhou Zhao, Zhu Zhang, Shuwen Xiao, Zhou Yu, Jun Yu, Deng Cai, Fei Wu, and Yueting Zhuang. Open-ended long-form video question answering via adaptive hierarchical reinforced networks. In *International Joint Conferences on Artificial Intelligence*, volume 3, page 4, 2018.

[340] Shuai Zheng, Kosta Ristovski, Ahmed Farahat, and Chetan Gupta. Long short-term memory network for remaining useful life estimation. In *ICPHM*, pages 88–95. IEEE, 2017.

[341] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. *National Conference of the American Association for Artificial Intelligence*, 34(07):13001–13008, 2020.

[342] Luowei Zhou, Jingjing Liu, Yu Cheng, Zhe Gan, and Lei Zhang. Cupid: Adaptive curation of pre-training data for video-and-language representation learning. *arXiv preprint arXiv:2104.00285*, 2021.

[343] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *National Conference of the American Association for Artificial Intelligence*, pages 7590–7598, 2018.

[344] Ren Zhou, Wang Xiaoyu, Zhang Ning, Lv Xutao, and Li Li-Jia. Deep Reinforcement Learning-based Image Captioning with Embedding Reward. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1151—1159, 2017.

[345] Yapeng Zhou and Miaohua Huang. Lithium-ion batteries remaining useful life prediction based on a mixture of empirical mode decomposition and arima model. *Microelectronics Reliability*, 65:265–273, 2016.

[346] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, pages 597–613, 2016.

[347] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *IEEE/CVF International Conference on Computer Vision*, pages 2242–2251, 2017.

[348] Linchao Zhu, Zhongwen Xu, Yi Yang, and Alexander G Hauptmann. Uncovering the temporal context for video question answering. *International Journal of Computer Vision*, 124(3):409–421, 2017.

[349] Linchao Zhu and Yi Yang. Actbert: Learning global-local video-text representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8746–8755, 2020.

[350] Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang. Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5802–5810, 2019.