Multi-Modal Learning for Video Understanding

Valentin Gabeur

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

École doctorale : MSTII - Mathématiques, Sciences et technologies de l'information, Informatique
Spécialité : Informatique
Unité de recherche : Laboratoire Jean Kuntzmann

Apprentissage multi-modal pour la compréhension de vidéos

Multi-Modal Learning for Video Understanding

Présentée par :

Valentin GABEUR

Direction de thèse :

Karteek ALAHARI Chargé de recherche HDR, INRIA CENTRE GRENOBLE-RHONE-ALPES	Directeur de thèse
Cordelia SCHMID Directeur de recherche, INRIA CENTRE DE PARIS	Co-directrice de thèse

Rapporteurs :

JOSEF SIVIC
Directeur de recherche, INRIA CENTRE DE PARIS
ANDREA VEDALDI
Professeur, University of Oxford

Thèse soutenue publiquement le **5 octobre 2022**, devant le jury composé de :

KARTEEK ALAHARI Chargé de recherche HDR, INRIA CENTRE GRENOBLE-RHONE-ALPES	Directeur de thèse
JOSEF SIVIC Directeur de recherche, INRIA CENTRE DE PARIS	Rapporteur
ANDREA VEDALDI Professeur, University of Oxford	Rapporteur
ERIC GAUSSIER Professeur des Universités, UNIVERSITE GRENOBLE ALPES	Président
FLORIAN METZE Professeur associé, Carnegie Mellon University	Examinateur
CORDELIA SCHMID Directeur de recherche, INRIA CENTRE DE PARIS	Co-directrice de thèse

Abstract

With the ever-increasing consumption of audio-visual media on the internet, video understanding has become an important problem in order to provide users with the right content. Compared to more traditional media, the video signal is inherently multi-modal, its information is scattered across several modalities such as speech, audio or vision. This thesis aims at designing and training deep learning models that exploit those different modalities to automatically understand video content.

While the community has developed high performance deep-learning models for uni-modal problems, processing multi-modal inputs such as video has received less attention. In the first part of this thesis, we introduce a transformer-based architecture that leverages pre-extracted unimodal features obtained at different moments in the video and fuses them into a single video representation. Taking advantage of the attention mechanism, we show that our model processes the video signal across both modalities and time. We couple our video encoder with a caption encoder into a cross-modal architecture and obtain state-of-the art results for the text-to-video retrieval task on three datasets.

Training such a model unfortunately requires a large amount of manually captioned videos. In order to leverage the billions of unlabelled videos on the internet, the common approach has been to use the accompanying speech as supervision to pretrain a video encoder on the other modalities. While the resulting encoder is capable of processing visual information and sound, it has not been trained to attend the spoken language in videos. In the second part of this thesis, we propose a method to pretrain a multimodal encoder on all video modalities, including speech. At each training batch, we completely mask out a different modality from the video encoder input and use it as supervision. We finetune our model on the task of text-to-video retrieval and show that our approach is particularly suitable for datasets where user queries relate to the spoken content in the video.

Extracting spoken language from the audio signal can be challenging, particularly when the audio is noisy. In the case of audio-visual media, the visual modality can provide precious cues to better extract speech from videos. In the third part of this thesis we introduce an encoder-decoder architecture as well as a pretraining strategy to not only train a speech recognition model on the audio signal but also on the visual signal. We evaluate the contribution of our approach on a challenging test set which demonstrates the contribution of the visual modality under challenging audio conditions.

Keywords: video, multi-modal, audio-visual, video retrieval, speech recognition, computer vision, deep learning, artificial intelligence.

Résumé

Avec la consommation toujours croissante de contenu audiovisuel sur Internet, la compréhension automatique de vidéos est devenue un problème important afin de proposer aux utilisateurs le contenu qui correspond à leurs attentes. Comparé aux médias plus traditionnels, le signal vidéo est intrinsèquement multimodal, ses informations sont réparties entre plusieurs modalités telles que la parole, l'audio ou la vision. Cette thèse vise à concevoir et entraîner des modèles d'apprentissage profond capables d'exploiter ces différentes modalités pour comprendre automatiquement le contenu vidéo.

Bien que des modèles d'apprentissage particulièrement performants aient été developpés pour les problèmes unimodaux, le traitement des données multimodales telles que la vidéo a reçu comparativement moins d'attention. Dans la première partie de ce manuscrit, nous introduisons une architecture basée sur un transformer qui exploite des descripteurs unimodaux pré-extraits à différents moments de la vidéo et les fusionne en une unique représentation. Bénéficiant du mécanisme d'attention, nous montrons que notre modèle est capable de traiter le signal vidéo à travers ses différentes modalités ainsi que temporellement. Nous couplons notre encodeur vidéo avec un encodeur de texte dans une architecture intermodale et établissons un nouvel état de l'art pour la tâche de recherche texte-vidéo sur trois jeux de données.

L'entraînement d'un tel modèle nécessite cependant une grande quantité de vidéos manuellement annotées. Afin de tirer parti des milliards de vidéos non annotées disponibles sur Internet, l'approche courante a été d'utiliser les mots prononcés dans la séquence comme supervision pour pré-entraîner un encodeur vidéo sur les autres modalités. Bien que l'encodeur résultant soit capable de traiter les informations visuelles et sonores, il n'a pas été entraîné à exploiter le discours oral des vidéos. Dans la deuxième partie de ce manuscrit, nous proposons une méthode pour pré-entraîner un encodeur multimodal sur toutes les modalités vidéo, y compris la parole. A chaque étape d'entraînement, nous masquons entièrement une modalité différente de l'entrée de l'encodeur et l'utilisons comme supervision. Nous affinons notre modèle sur la tâche de recherche de vidéos et montrons que notre approche est particulièrement adaptée aux jeux de données où les requêtes des utilisateurs concernent le discours prononcé dans la vidéo.

Extraire la transcription du signal audio peut être difficile, en particulier lorsqu'il est bruité. Dans le cas d'un contenu audiovisuel, la modalité visuelle peut fournir des indices précieux pour mieux extraire

la parole des vidéos. Dans la troisième partie de ce manuscrit, nous introduisons une architecture encodeur-décodeur ainsi qu'une stratégie de pré-apprentissage pour entraîner un modèle de reconnaissance vocale non seulement sur le signal audio, mais également sur le signal visuel. Nous évaluons la contribution de notre approche sur un jeu de données de test qui démontre la contribution de la modalité visuelle pour la reconnaissance de la parole dans des conditions audio difficiles.

Mots-clés: vidéo, multi-modal, audio-visuel, recherche de vidéos, reconnaissance automatique de la parole, vision par ordinateur, apprentissage profond, intelligence artifielle.

Acknowledgements

This work was made possible thanks to the help of many people. First and foremost, I would like to thank my supervisors, Cordelia Schmid and Karteek Alahari, for their encouragement and guidance throughout the course of my PhD. Thank you for giving me such a great opportunity and encouraging me to never give up.

I have also had the pleasure of collaborating with other amazing co-authors including Chen Sun, Arsha Nagrani, Paul Hongsuck Seo, Xavier Martin, Jean-Sébastien Franco and Grégory Rogez, to whom I am forever grateful for their huge contribution in my research. Special thanks to Grégory and Jean-Sébastien for believing in me from early on, before I had conducted any research.

I would like to thank the members of the jury for accepting to evaluate my work and especially to Andrea Vedaldi and Josef Sivic for taking the time to review this manuscript.

Going back to school to study machine learning after having worked for six years as a mechanical engineer was one of the best decisions of my life, but it was also one of the hardest. I am grateful to Viviane Cadenat for allowing this career change by accepting my application to the robotics master at Toulouse University.

I am thankful to the authors of "Use What You Have: Video Retrieval Using Representations From Collaborative Experts" for sharing their codebase and features, and Samuel Albanie, in particular, for his help with implementation details. I would also like to express my gratitude to the organisers of the CVPR 2020 Video Pentathlon challenge for setting up such an exciting competition.

To my amazing colleagues at Inria Thoth and Google Ganesha, thanks for the many interesting and enjoyable discussions around coffee. I am especially grateful to Alexander for proofreading this manuscript and Nathalie for helping me deal with the administrative tasks.

Last but not least, I would like to thank my friends and family, especially Mayra and Emilio for their unconditional love and support.

Contents

Chapter 1

Introduction

Our society is increasingly becoming digitized. We communicate through e-mails, read e-books and buy on e-commerce platforms. Digital content usage is not restricted to large desktop computers anymore, it is now accessible everywhere through 'smartphones' that keep us connected to the entire world.

These smartphones we own do not only allow us to consume more digital content, they also empower us as content creators. Thanks to the embedded digital camera and microphone we carry in our pocket at all times, anybody can grab a scene they find important, be it a private family moment or an event to be shared with the whole world. As a result, more than 500 hours of video are now uploaded to the internet every minute[1].

Some of these videos are potentially undesirable (violence, privacy violation, pornography...), and therefore need to be moderated before being made publicly available. Unless efficiently searchable, videos are also effectively worthless: they have to be properly indexed to be retrievable among billions of other videos at sub-second latency. At such a scale and time constraints, these operations cannot be performed manually. In order to automate these tasks, algorithms require low-dimensional representations of videos that accurately summarize their content.

These video representations used to be primarily based on the keywords uploaded by the content creators. As malicious uses have become increasingly common on the internet, we cannot trust such information anymore. For example, a user uploading a promotional video could deliberately describe it with wrong keywords just to draw more viewers to it. Objective descriptors are therefore required to accurately describe video content.

[1] https://www.statista.com/statistics/259477/hours-of-video-uploaded-to-youtube-every-minute/

The machine learning community has proposed many techniques to compute compact representations of data, but they have so far mostly focused on uni-modal problems: designing image descriptors, extracting speech from audio, learning language representations, performing audio classification, etc.

Video data, however, gathers information coming from several sources. A video scene is captured using two different but synchronized hardware: a microphone and a camera. While the signals captured by these two sensors both have a temporal dimension, their structure is very different. Once digitized, a monophonic audio signal is composed of one value per timestep, but *thousands timesteps* per second. On the contrary, a standard camera captures about 30 frames per second, but these are high dimensional and *organized in a 2D structure*.

Within those visual and audio signals comes information of very different types. In the visual signal, information can be extracted from the appearance of objects, characters and backgrounds. Luminosity can give hints about the time of the day at which a scene has been captured. There might be readable words or symbols from the scene itself or embedded as overlaid text. The relative motion of the subjects between frames tell us about the actions being performed and at which speed. In the audio signal, noises can indicate the use of specific tools or materials. Sounds of animals can be heard, different types of music can be played. The analysis of spoken language opens up a whole new dimension of information as it tells us about the characters opinions, feelings or future plans. And there is not only meaning in what is said but also how it is said (prosody).

In this manuscript, we refer to these different information sources in the video as 'modalities'. We consider as modalities all the possible representations of the video signal. Since there are as many modalities as approaches to extract video semantics, there are many possible ways to decompose the video signal into modalities. We present a possible decomposition in figure 1.1.

In this thesis, we will study how to best combine these modalities information to obtain a multi-modal representation of a video. We will show that the meaning of a video is more than the concatenation of these modalities semantics and that there are also valuable cross-modal cues lying at their intersection (Fig. 1.2).

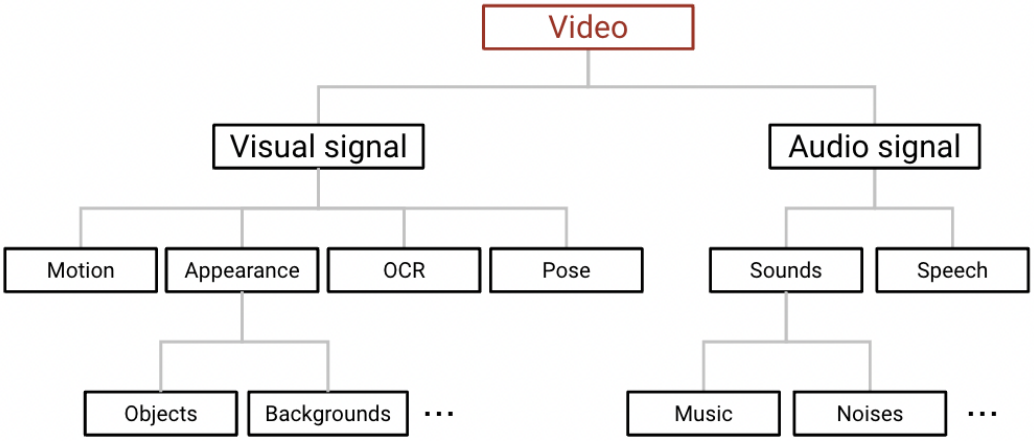

Figure 1.1: Example of a hierarchical decomposition of the video modalities. Other decompositions are possible depending on the approaches chosen to extract each modality representation.

1.1 Goals and Challenges

In this thesis, we aim at designing and training deep learning models capable of multi-modal video understanding. For instance, when provided with both the sequence of visual frames and the audio track of a video, we would like to automatically extract the semantics of the video, i.e., which information in the video would be valuable for a human if they were to describe it. Given a video, our model should therefore be able to understand what actions are being performed, in which order, what is being said, what is the source of the sounds, where is the scene taking place, what emotions are expressed by the characters, why it is funny, etc.

We will evaluate the video understanding capabilities of our models on two tasks: text-to-video retrieval and audio-visual speech recognition. In the remainder of this section, we will introduce these tasks and their associated challenges.

1.1.1 Text-to-video retrieval

Text-to-video retrieval is the task of matching a text query with its most relevant candidates in a video collection. It is a typical task of a video search engine where a user enters a text query to retrieve relevant videos. More precisely, for a given query, the task consists in estimating the similarity between the query and all the videos of the dataset, and then ranking the videos according to their similarity with the query. The evaluation of a

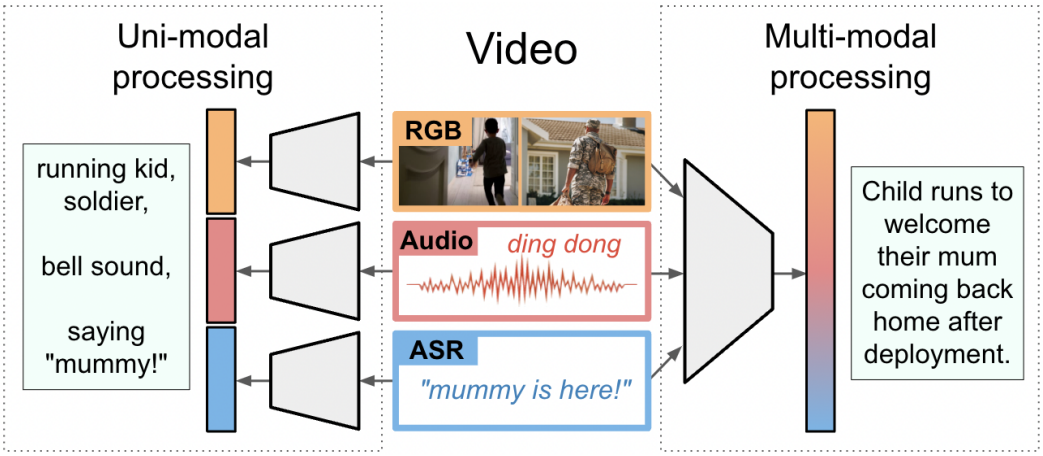

Figure 1.2: Considered independently, video modalities only provide a limited representation of a video scene (left). On the contrary, jointly processing video modalities can extract the multi-modal cues necessary to the global understanding of the video (right).

model's performance at text-to-video retrieval is typically performed on a dataset of videos that have been manually annotated with one or more captions (Fig. 1.3).

One of the first challenges for video retrieval is scale. The video collection to retrieve from may contain billions of videos while the best video candidates should be provided to the user in less than a second. Such a constraint forbids similarity computation from scratch on raw video signal for each user query but instead pushes for offline computation of video representations that could be re-used for each query. Mapping text and video to a joint embedding space allows computation of a similarity score via a single dot product but gives up retrieval accuracy for retrieval speed [101].

A challenging specificity of video media lies in its multi-modal nature: There is information in the spoken sentences, the sounds heard, the visible objects, the actions performed, the face expressions, the words appearing on screen, etc. All those sources of information complement each other and we should therefore not only encode information from those individual video modalities but also information that lies at their intersection.

Another challenging aspect of video is its temporality. It is important to pay attention to the relative order of the events in a video to understand a scene. Videos can also vary greatly in duration. It is necessary to encode videos of variable durations into a fixed-size representation without discarding the temporal information.

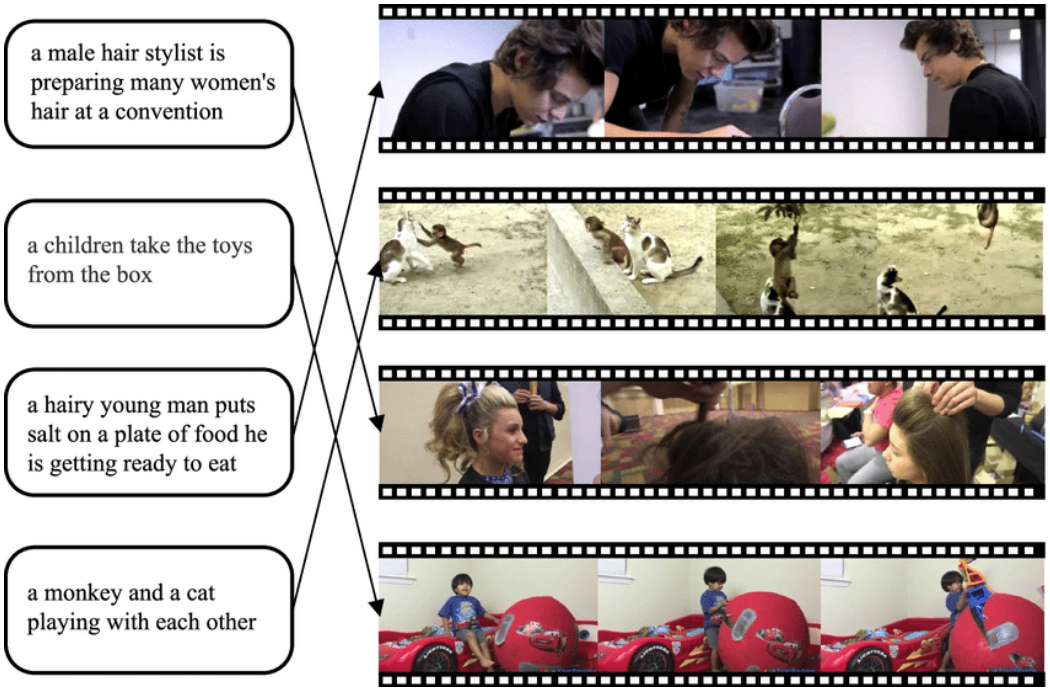

Figure 1.3: Text-to-video retrieval consists in matching textual queries with the most relevant videos. Performance is typically evaluated on a dataset of video-caption pairs where each caption should be matched with the video it describes. (Image courtesy of Lin et al. [89]).

Even if highly dissimilar in nature, both text queries and video candidates need to be encoded in a common embedding space in order to efficiently perform text-to-video retrieval. But jointly training a query encoder and a video encoder on this task requires a large dataset of video-captions pairs which is both difficult and expensive to annotate. Self-supervised approaches [7, 102, 104] have proposed to leverage the massive amount of unnannotated videos available on the internet to pretrain encoders by using the speech information in videos as 'pseudo captions'. However, the video in that case is stripped from the audio and only the visual signal is provided to the video encoder. This results in the inability of the video encoder to process the audio modality.

1.1.2 Audio-visual speech recognition

The second problem studied in this dissertation is audio-visual automatic speech recognition (AV-ASR), the task of extracting spoken language from

a video. Automatically producing speech transcriptions from video media has a wide range of applications, the most obvious one certainly being accessibility to people with hearing deficiencies. Speech transcript estimation is also often used as an intermediate step [15] or a support task [83] for speech translation, which in the case of video aims at automatically producing subtitles [134]. Furthermore, as we will show in Chapter 4, speech is an essential modality to exploit for video understanding, its proper estimation from the audio-visual signal is therefore paramount. AV-ASR methods build on the large amount of work on automatic speech recognition (ASR), the task of speech recognition on audio-only signal. Most recents approaches to speech recognition [160, 169] obtain word error rates (WER) of less than 2% on Librispeech, a popular audiobooks dataset. On more "in the wild" datasets such as How2 [134], the performance is much worse, which shows that ASR is still far from being solved.

One of the first possible issues when performing speech recognition is the quality of the audio signal itself, recorded using a low quality microphone or an analog format for example. Audio noise may also come from the recording environment: The audio scene could be happening near noisy machines, wind could be blowing in the microphone or the speaker could be so far from the microphone that ambient noise is exacerbated. Music or baby cries can also overlap with the speech to be transcribed.

The variability in speaker elocution itself is another challenging aspect. Speakers are sometimes submerged with emotions and difficult to understand. Some speak so fast to the point of being incomprehensible. Many languages feature strong regional variations where the same word can be pronounced in different ways. There are thousands of languages and an infinite number of words, rarely used words combinations can be easily mistaken with more common ones of similar pronunciation. This is especially true for proper nouns, which can be the name of a particular person, place, thing, or even a brand.

Fortunately, there is an opportunity to overcome some of the previously mentioned challenges using cues from different modalities in the visual signal. The lip movements of the speaker can reveal the phonemes that they are articulating. Persons, places, objects, actions or brands that the speaker is referring to might be visually recognisable on screen, therefore helping their correct transcription. Text can be present in the scene, or might even be overlaid on the image, and provide the correct spelling of some words (Fig. 1.4).

It is certainly easier to understand what is being talked about if we are provided with more context. It is true for the spoken context, i.e., what has

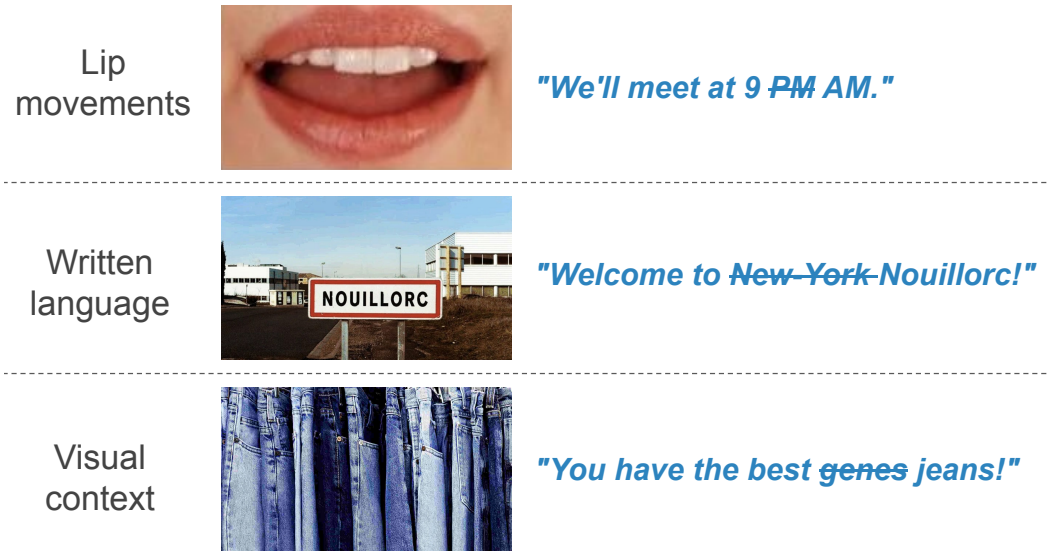

Figure 1.4: The visual modality can provide strong cues to better recognize speech in videos.

been said previously, but also for the visual context, i.e., what is being seen in the video. Unfortunately, the spoken context is often limited by the acceptable input size of a model and the video duration. On the contrary, the visual context is temporally aligned with the speech, and therefore always accessible.

But using the visual modality to improve speech recognition comes with its own challenges. Visual information comes in different forms that require specialized models for their extraction. Processing speakers lip movements requires a high framerate over a small zone of the image (the speakers lips) that therefore needs to be high resolution. Recognising backgrounds and actions can be performed at a lower frame rate but needs to be run on the whole image. Leveraging text in the image requires optical character recognition and natural language processing to understand its semantics.

It is unclear how this visual information coming from multiple modalities should be merged with the audio signal itself to extract cross-modal semantics. The ASR task is dominated by the audio modality and it is challenging for an AV-ASR model to learn from the visual signal when the audio signal only is sufficient for most examples.

Finally, the high computational cost of video processing constitutes a challenge to AV-ASR. Both audio and visual raw signals are very high dimensional and therefore difficult to process jointly.

1.2 Contributions

1.2.1 Modality fusion with the multi-modal transformer

Most of the existing methods for the text-to-video retrieval problem do not fully exploit cross-modal cues present in video. Furthermore, they aggregate per-frame visual features with limited or no temporal information. Our first contribution is a multi-modal transformer to jointly encode the different modalities in video, which allows each of them to attend to the others. We also leverage the transformer architecture to encode and model the temporal information. On the natural language side, we investigate the best practices to jointly optimize a caption encoder together with the multi-modal transformer (Fig. 1.5). This novel framework allows us to establish state-of-the-art results for video retrieval on three datasets. We also obtained the first place at the CVPR 2020 video pentathlon challenge [8]. We present the details of this cross-modal architecture in Chapter 3.

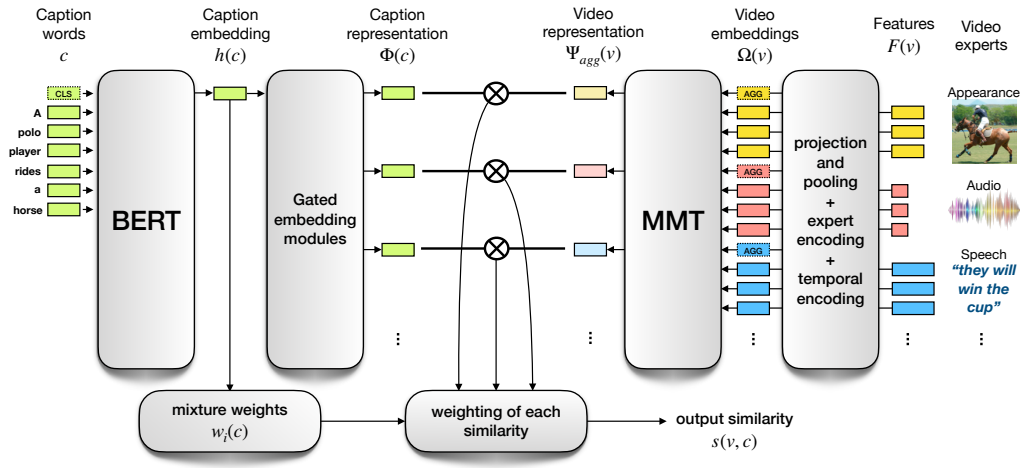

Figure 1.5: Our cross-modal framework for text-video similarity estimation. We use our Multi-modal Transformer (MMT, right) to encode video, and BERT (left) for text.

1.2.2 Multi-modal pretraining of a video encoder

Pretraining on large-scale unlabelled datasets has shown impressive performance improvements in the fields of computer vision and natural language processing. Given the advent of large-scale instructional video

datasets, a common strategy for pretraining video encoders is to use the accompanying speech as weak supervision. However, as speech is used to supervise the pretraining, it is never seen by the video encoder, which does not learn to process that modality. We address this drawback of current pretraining methods, which fail to exploit the rich cues in spoken language. The second contribution of this thesis is a self-supervised approach to pretrain a video encoder using three different video modalities as supervision, namely, appearance, sound, and transcribed speech. At each training batch, we mask an entire modality in the input and predict it using the other two modalities (Fig 1.6). This encourages each modality to collaborate with the others, and our video encoder learns to process appearance and audio as well as speech. We show the superior performance of our 'modality masking' pretraining approach for video retrieval on the How2R, YouCook2 and Condensed Movies datasets. We give the details of our approach in Chapter 4.

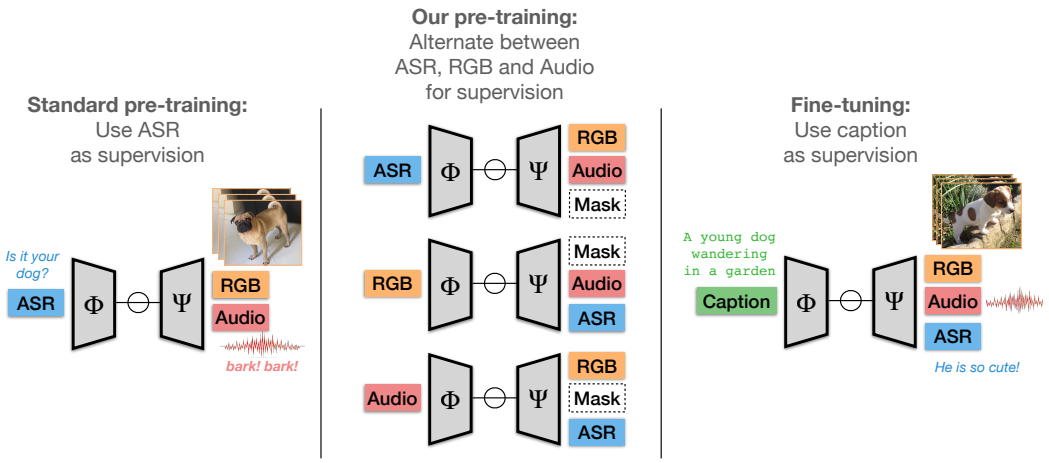

Figure 1.6: A common paradigm in learning from instructional videos is use transcribed speech (from ASR) (projected here using an encoder Φ) to supervise a video encoder Ψ (left). Instead, we train our video encoder Ψ with three inputs – RGB, audio and transcribed speech (ASR), and alternate between masking and predicting an entire modality at a time (middle). At the time of finetuning (right), our video encoder has been pretrained to use all video modalities.

1.2.3 Visual context for improved speech recognition

Unlike AV-ASR works that simply focus on the lip motion, we investigate the contribution of entire visual frames (visual actions, objects, background etc.). This is particularly useful for unconstrained videos, where the speaker is not necessarily visible. To solve this task, we propose a new sequence-to-sequence AudioVisual ASR TrAnsformeR (AVATAR) which is trained end-to-end from spectrograms and full-frame RGB (Fig. 1.7). To prevent the audio stream from dominating training, we propose different word-masking strategies, thereby encouraging our model to pay attention to the visual stream. We demonstrate the contribution of the visual modality on the How2 AV-ASR benchmark, especially in the presence of simulated noise, and show that our model outperforms all other prior work by a large margin. Finally, we also create a new, *real-world* test bed for AV-ASR called VisSpeech, which demonstrates the contribution of the visual modality under challenging audio conditions. We explain the details of our approach in Chapter 5.

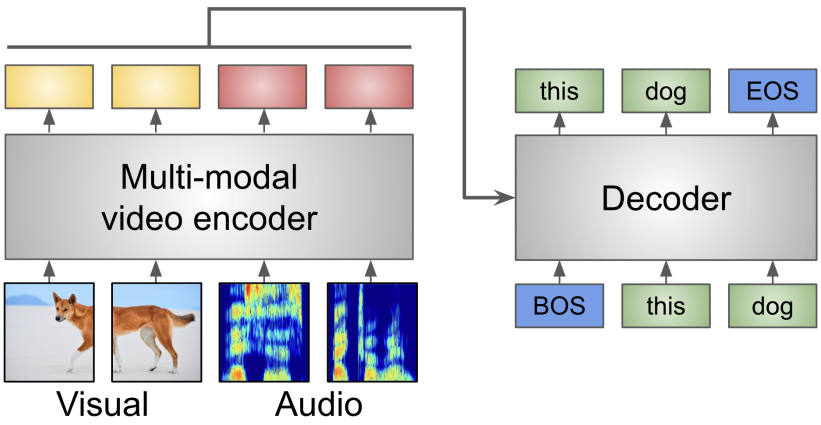

Figure 1.7: **AVATAR:** We propose a Seq2Seq architecture for audio-visual speech recognition. Our model is trained end-to-end from RGB pixels and audio spectrograms to estimate the speech in a video.

1.3 Publications

This manuscript is based on material published in the following papers:

- [50]: Valentin Gabeur, Chen Sun, Karteek Alahari and Cordelia Schmid. *Multi-modal Transformer for Video Retrieval.* ECCV 2020 (Spotlight) - Chapter 3

- [47]: Valentin Gabeur, Arsha Nagrani, Chen Sun, Karteek Alahari and Cordelia Schmid. *Masking Modalities for Cross-modal Video Retrieval.* WACV 2022 - Chapter 4

- [48]: Valentin Gabeur, Paul Hongsuck Seo, Arsha Nagrani, Chen Sun, Karteek Alahari and Cordelia Schmid. *AVATAR: Unconstrained Audiovisual Speech Recognition.* INTERSPEECH 2022 - Chapter 5

Our Multi-modal Transformer for Video Retrieval won the first place at the CVPR 2020 video pentathlon challenge [8]:

- [49]: Valentin Gabeur, Chen Sun, Karteek Alahari and Cordelia Schmid. *CVPR 2020 video pentathlon challenge: Multi-modal transformer for video retrieval.* CVPR 2020 Video Pentathlon Workshop

This manuscript does not include material published by the author in the following paper:

- [46]: Valentin Gabeur, Jean-Sebastien Franco, Xavier Martin, Cordelia Schmid and Gregory Rogez. *Moulding Humans: Non-parametric 3D Human Shape Estimation from Single Images.* ICCV 2019

1.4 Software and datasets

The work conducted in this thesis has led to the following software and datasets:

- Multi-modal Transformer. Source code of the cross-modal architecture for video retrieval used in chapters 3 & 4.
  https://github.com/gabeur/mmt

- Pre-extracted features. Motion, audio, object, faces, scene, speech and OCR features extracted from the MSR-VTT, ActivityNet and LSMDC datasets. The features are used to reproduce the results reported in chapter 3.
  http://thoth.inrialpes.fr/research/video-features/

- VisSpeech dataset. A challenging AV-ASR test benchmark used to evaluate the performance of our AVATAR model in chapter 5.
  https://gabeur.github.io/avatar-visspeech

Chapter 2

Background

As this thesis tackles video understanding, it builds on numerous works that have investigated how to individually process the different modalities that compose a video. These modalities are indeed not specific to video and often constitute a media by themselves, with its own tasks of interest. For example, speech and sounds are also present in the radio media, still images also contain visual information, text documents are written language, etc. Another reason why video modalities have been independently studied is that some video tasks are by nature uni-modal and do not benefit from cross-modal queues. For example, human pose estimation is a visual task that would probably not benefit much from the audio modality.

In this chapter, we will provide the reader with some background on how to separately process the different modalities that constitute the video signal. We will focus on three main modalities:

- Vision, i.e. what is visible in the video.
- Audio, i.e. the sounds that can be heard.
- Language, either spoken or written.

More closely related to the contributions of this thesis, we will introduce in the next chapters the previous works that have looked into *jointly* processing these modalities to obtain multi-modal video representations. We will present related works on multi-modal fusion in Chapter 3, video and language pretraining in Chapter 4, and audio-visual speech recognition in Chapter 5.

2.1 Visual signal processing

Sight is one of the most important human senses as it allows us to perceive our 3D environment, recognize familiar faces and distant objects. Cameras are the tools we use to capture visual information from the environment and digitize it into pixel grids. In this section, we will describe how to automatically process this visual information.

2.1.1 Image representation

Image information fundamentally lies in the gradient of pixel intensity which is highest at corner, edges and high contrast zones of the 2D pixel grid. In order to capture the semantics of images into a higher level representation, the traditional approach has been to hand-design visual descriptors. The histogram of oriented gradients (HOG) [38] is a widely used visual descriptor. After computing the intensity gradient for each pixel of the image, the image is divided into blocks and an histogram of gradient directions is computed for each block. The final representation consists in the concatenation of the histograms of all blocks. The Scale Invariant Feature Transform (SIFT) [91, 92] is another popular descriptor. It computes difference of Gaussians at several scales to detect keypoints in the image. A feature is then associated to each keypoint based on gradient histograms of neighboring pixels.

In order to perform image classification, these image descriptors then need to be processed through a classifier. The k-nearest neighbors algorithm (k-NN) [36] is a non-parametric method to classify a sample by simply looking at the labels of its closest neighbors in feature space. While simple and intuitive, this method can become computationally inefficient on large datasets as it requires calculating the distance to all samples and keep them in memory. The Support Vector Machine (SVM) [35] has been another popular classification algorithm for computer vision. It leverages kernel methods to map an input space to an implicit high dimensional feature space where linear models can be trained. The SVM algorithm selects a set of support vectors to find a decision boundary with large margin in feature space.

After remaining ignored during many years as they require large training datasets and compute power, artificial neural networks (NNs) have recently received renewed interest. Inspired by biological neural networks, artificial neural networks were first introduced by Rosenblatt in 1958 with the Perceptron model [131]. The Perceptron, originally implemented as a

single layer, consisted in a linear transformation of its inputs followed by an activation function. Multilayer perceptrons [71] were later introduced as stacked perceptron layers. They were shown to be universal approximators, i.e., capable to approximate any function, under certain conditions [37]. The multilayer perceptron is considered to be the first deep learning architecture as its multiple layers progressively extract higher-level semantics from the input. In the case of image processing, convolutionnal neural networks (CNNs) have brought multiple advantages compared to standard NNs. The convolution operation allows for a reduction in model parameters while enforcing equivariance to input transformation. The first convolutional network was introduced by Fukushima in 1980 with the Neocognitron [45].

Theorized by Kelley in 1960, the back-propagation technique [77] was only popularized in 1986 by Rumelhart to train NNs [132]. By applying the chain rule, the prediction error is back-propagated through the model to obtain the gradient of the error relative to the parameters. The model parameters can then be iteratively optimized by gradient descent. Combining these advances in model architecture and training, Le Cun et al. released LeNet [84], a CNN designed to recognize handwritten digits. CNNs are trained end-to-end, meaning that all model weights are optimized using back-propagation, leaving only the architecture design as a manual task. Compared to handcrafted descriptors, CNNs find the optimal filters automatically. In a multilayer CNN, the first layers typically learn to identify low-level features such as edges so that the following layers can learn higher semantics such as recognizing a human face (Fig. 2.1).

Starting 2010, the advent of large scale datasets made room for deep learning approaches. The popularization of ImageNet [39] as the standard benchmark helped compare methods against each other on common ground. The 2012 ImageNet large scale visual recognition challenge (ILSVRC) [133] has seen a turning point in computer vision with the introduction of AlexNet [81], the first deep learning model surpassing traditional approaches on image classification. AlexNet was one of the first implementation of a neural network to leverage parallel computation on graphics processing units (GPUs). The CNNs were made deeper by stacking more layers [144] and introducing skip connections in residual blocks [64]. Since then, the computer vision field has been dominated by deep learning approaches.

As an alternative to CNNs, the Transformer architecture, originally designed for language processing, has recently been adapted to computer

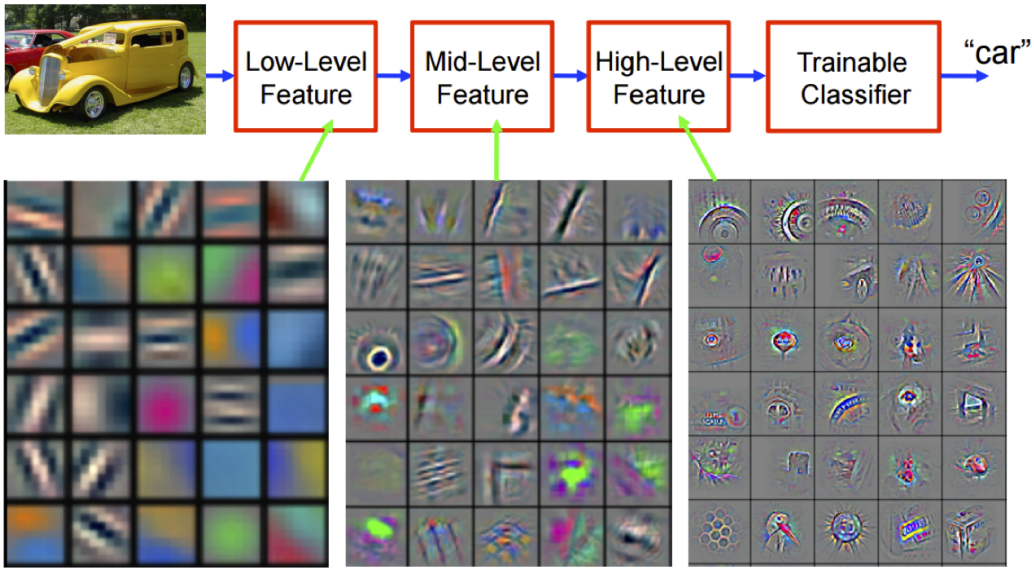

Figure 2.1: Visualization of a trained CNN features. The reconstructions correspond to feature maps projected to pixel space using the deconvolutional network approach [165]. (Image courtesy of Zeiler and Fergus [165])

vision. It takes as input a set of embeddings and progressively contextualize them by letting each embedding attend to all the others. Because it uses self-attention (global operation) instead of convolutions (local operations), the transformer model has less inductive biases than CNNs. We explain the details of the transformer architecture in section 2.3.2. Diverse strategies have been used to have a transformer process pixels directly with a reasonable computational cost: Diverse strategies have been adopted to process pixels with Transformer at a reasonable computational cost: With the Vision Transformer (ViT) [42], the image is split into patches and each patch is linearly transformed into an input embedding. The Perceiver [72] is able to directly cross-attend pixels by iteratively refining a limited number of latent embeddings.

As models capacity has continuously increased, as well has the need for annotated data. But annotating large image datasets is expensive and difficult, especially for specialized content like medical images or satellite images. Researchers have therefore tried to leverage large unlabeled image collections to learn meaningful visual representations in a self-supervised way, meaning that both the inputs and labels are derived from an unannotated dataset. Several pretext tasks have been proposed to obtain the pseudo labels, for example rotation prediction [53], image

colorization [167] or context prediction [41]. Pseudo labels have also been obtained iteratively through clustering [24] and online using learned prototypes [25]. Enforcing invariance of representation to image augmentations (e.g., cropping, rotation, color jittering) is key to many image self-supervised approaches, building on this objective, self-supervised contrastive methods have recently shown great success to learn visual representations [31, 63]. They consist in pulling together in embedding space representations of a same image (positive pairs) and pushing away from each other the representations of different images (negative pairs). Some recent methods [19, 32, 59] differ from contrastive learning as they only rely on positive pairs to learn an image representation but require enforcing additional constraints to avoid collapse (i.e., when the encoder outputs a constant vector, independent of the inputs).

2.1.2 Video representation

In this section, we describe methods for obtaining a representation from visual-only video media, i.e., a sequence of images. We will particularly focus on the action recognition task as it is the main benchmark for video models evaluation.

Because of the similarity between images and videos, video processing methods have been greatly inspired by image representation works introduced in the previous section and have followed the same evolution, from manually designed descriptors to end-to-end trained CNNs.

Compared to static images, videos provide additional information by adding a temporal dimension, therefore providing short term motion information as well as longer term scene transitions. Processing video is also much more computationally expensive than processing static frames, as it is composed of many of them. The video signal is also highly redundant, with neighboring frames often being very similar to each other.

In order to process the additional temporal dimension, 2D image CNNs have been extended to 3D video CNNs [74]. This computationally expensive approach has been called "Slow fusion" in subsequent work [76] as temporal and spatial information are progressively fused at each CNN layer of the model. Karpathy et al. [76] make the surprising observation that on the action recognition task, spatio-temporal approaches only provide modest performance improvement compared to a single frame CNN. The authors hypothesize that motion information is either not critical or difficult to learn and process by the model.

To ease its processing, motion information can be explicitly provided to a model through optical flow. Optical flow is the apparent motion of objects in a visual scene. Computed from two consecutive video frames, it is encoded as a motion vector for each pixel. Simonyan and Zisserman [143] leverage optical flow to explicitly provide motion information to a two stream CNN. One stream extracts spatial information from RGB input while the other stream computes temporal information from multi-frame optical flow. Ng et al. [114] extend this approach to process long videos (up to two minutes). To do so, they propose to only process one RGB frame per second and encode the motion through optical flow. They employ a recurrent network to model the dynamics of the video. Carreira and Zisserman [26] compare the previously proposed video encoder architectures and propose a two-stream 3D CNN. Their model processes sequences of optical flow and rgb frames separately and then merge the predictions. They demonstrate the advantage of inflating the 2D filters of image-pretrained 2D CNN to initialize the filters of both 3D CNNs.

SlowFast Networks [44] also feature two parallel streams, but do not use optical flow as only RGB frames are provided as input. One heavyweight stream operates at a low frame rate to capture spatial semantics while another lightweight stream operates at high frame rate to capture motion. Both streams exchange information through cross-streams connections. To reduce the computational burden of 3D convolutions, the S3D network [156] introduces the separable 3D convolution as a 2D spatial convolution followed by a 1D temporal convolution.

Similar to image processing, the transformer architecture has successfully been applied to video processing. With VideoBert [148] and CBT [147], short range video features are first extracted using S3D and then provided as input to a transformer to extract their long-range temporal evolution. The transformer architecture has also been applied end-to-end to video voxels with ViViT [11] which leverages the pretrained weights of image transformers [42] to initialize the video transformer and employs 3D Tubelet embeddings (Fig. 2.2) as an extension to 2D embeddings.

While large image dataset annotation is already difficult and expensive, the added temporal dimension makes video dataset annotation even more challenging. Researchers have therefore looked at exploiting the vast amount of unannotated videos available on the internet through self-supervised learning. Similar to still-image self-supervised approaches, Patrick et al. [121] tackle video representation learning by enforcing invariance to random spatial cropping augmentations, which

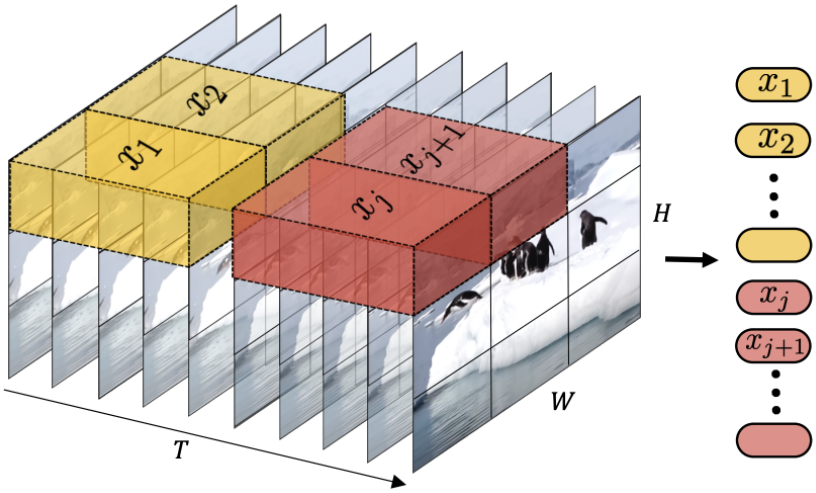

Figure 2.2: Tubelet embeddings. The 2D spatial patches are extended in the temporal dimension to form 3D tubelets that span several video frames. The pixels of each tubelet are flatten and lineraly embedded into a single embedding per tubelet. (Image courtesy of Arnab et al. [11])

are performed in feature space to reduce memory cost. Considering audio-visual content, the audio track accompanying a video constitutes a possible self-supervision to train a video model. Asano et al. [12] have successfully exploited this modality for unsupervised video dataset labelling and video representation learning. In addition to mere audio sounds, the spoken language potentially present in the audio track itself can contain high level semantics for representation learning. Compared to image self-supervised approaches that typically rely on augmentations of a same image to obtain positive pairs, pseudo labels for video can be extracted from the speech itself without relying on artificial visual augmentations. But these freely available pseudo-labels are also very noisy as what is said does not necessarily correspond to what is seen in the video.

Instructional videos is one of the video category where the speech modality correlates the most with the visual modality. As their intent is to explain a complex task, the instructional video creators often explicitly describe their actions and the objects they are manipulating. This has led several works to explore the use of instructional videos for tasks like action steps alignment [96], task steps discovery [5], action segmentation [129] or audio-visual speech recognition [134]. But it is only with the open release of the HowTo100M dataset [104] that instructional videos have been used at massive scale for self-supervised learning. The original HowTo100M

pretraining approach consists in jointly training a visual model and a language model to embed videos and narrations in a common embedding space. The similarity between videos and narrations is maximized if issued from the same clip and minimized if sampled from different clips. Various contrastive objectives have been experimented to optimize the weights of video-language models such as triplet loss [104] or variants of the noise contrastive loss [7, 102].

2.2 Audio signal processing

Sound is a vibration that propagates as a longitudinal wave in a medium, usually air. Captured by a microphone, this change in air pressure is converted to an electrical signal that can be exploited as-is (analog) or discretized (numeric).

Sampled at 16kHz, one second of raw audio signal waveform consists in 16 thousand values. Observed independently, each of these do not carry any meaning. It is only when looking at hundred of these values together that we start to notice the ondulations of the waveform that characterize possible noises or frequencies. Digital audio in its raw form is therefore challenging to exploit by a learning algorithm, it is usually first preprocessed using human-designed representations that capture its essential features before being used for sound classification or speech recognition.

2.2.1 Audio representation

Audio signal representation has historically been approached using a set of manually designed low level descriptors (LLDs). The Zero-Crossing Rate (ZCR) is one of the simplest, it corresponds to the rate of sign-changes of the signal during an audio clip and can be interpreted as a measure of the noisiness of a signal. The energy contour is another popular LLD. It is obtained by summing the squared samples of an audio segment. Because of the poor representation power of those LLDs and the high temporal variability of the audio signal, the second order statistics of the LLDs were shown to provide better representations that the LLDs themselves [136].

More advanced audio representations have built on the spectral representation of sound and its decomposition into sine waves, varying in amplitude, frequency and phase. While it has been shown that the human ear is almost insensible to variations in phase [65], the amplitude (loudness) and frequency (pitch) are of high importance to the human perception of sound.

Mathematically, the decomposition of raw audio signal can be achieved through Fourier analysis. The Fourier Transform is used to decompose a waveform into a set of sine waves, characterized by their amplitude and frequency (Fig. 2.3). Because of the temporal variability of the audio signal, the Fourier transform is applied to short overlapping segments of audio. Those segments are smoothed with a Hamming window to avoid Gibbs phenomenon oscillations caused by signal discontinuities. The Fourier transform transfers the audio information from the time domain to the frequency domain. It produces a frequency spectrum for each audio segment. Building on this spectral representation of the audio signal, researchers have designed frequency-based LLDs such as spectral centroid [136], signal bandwidth and pitch [99].

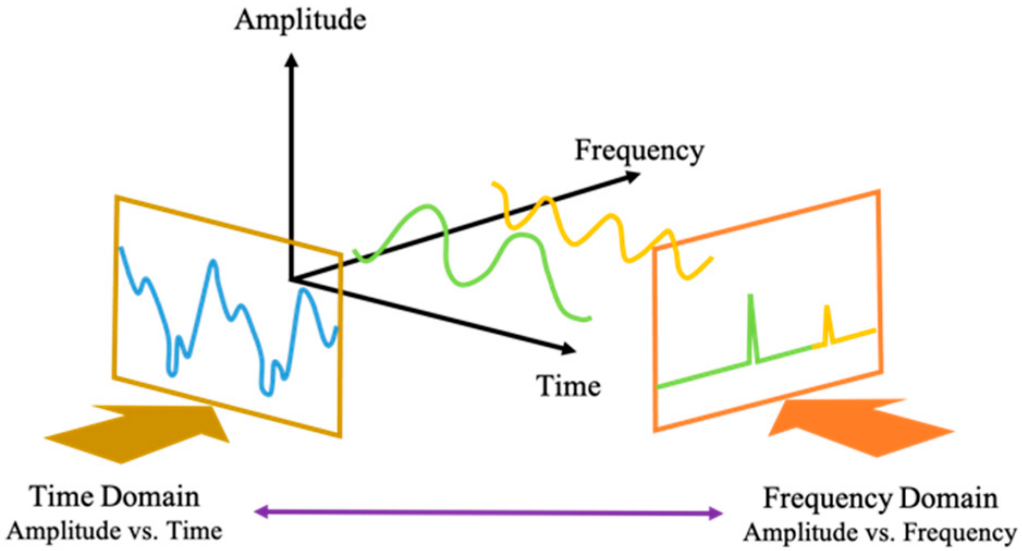

Figure 2.3: Decomposition of a waveform into a set of sine waves with different amplitudes and frequencies. (Image courtesy of Purnomo et al. [126]).

While the previously introduced low-level descriptors can be successfully used to discriminate different types of audio signal, their representation power is limited in the case of more complex tasks such as speech recognition. Speech is a highly dynamic signal at the macro level, the succession between consonants and vowels forms a sequence of phonemes carrying the semantics of a speaker's message. But if observed at a lower level, for the duration of a phoneme, the sound appears static. At subsecond scales, the speech signal usually varies slow enough that its acoustic features can be considered constant. Nakajima et al. [113] showed that

speech cut into short static segments was almost perfectly intelligible with a temporal resolution of 40 ms or less, which allows audio signal representation as a sequence of regularly extracted audio features.

Inspired by the physiological mechanism for sound perception, mel filterbanks are a popular method for computation of such audio features. They aim at discretizing the continuous frequency spectrums into a compact representation. The frequency spectrum is processed through mel-spaced triangular filters. The log-power of the filtered signal is calculated to obtain a value for each mel-frequency, hence transforming the frequency spectrum of a short segment into a feature vector. The 1D audio waveform becomes a 2D audio spectrogram (Figure 2.4).

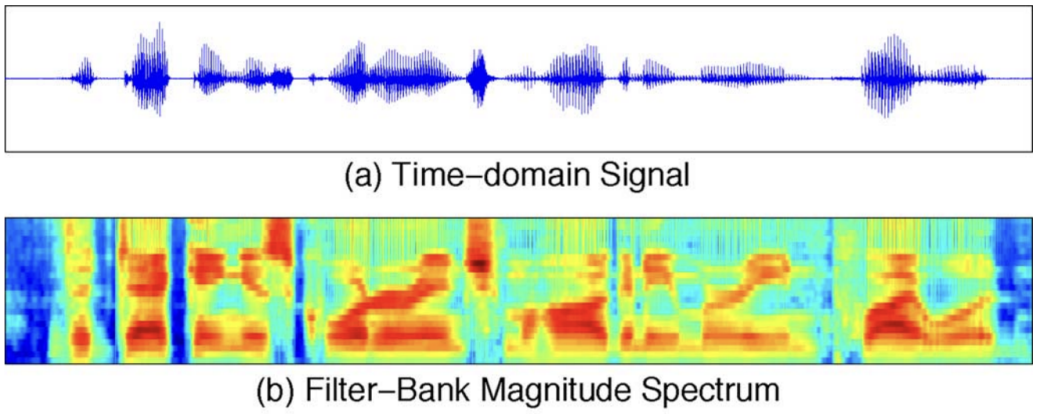

Figure 2.4: A raw audio waveform (a), and its corresponding spectrogram of mel filterbanks features (b). (Image courtesy of McCowan et al. [98]).

More recent approaches have experimented with learning an audio frontend competitive with mel filterbanks [164] or learning directly from raw audio waveform [4] but obtain limited gain compared to using mel filterbanks.

2.2.2 Sound recognition

Sound recognition is the task of identifying the different categories of an audio segment, e.g. music, speech, noise. It can be performed at different hierarchical levels of detail, e.g. what kind of music is played, by what instrument, what is the noise origin.

One of the first application of audio classification has been to discriminate between speech and music, two of the most common content in audio. In [135], short segments of broadcast FM radio are classified between music

and commercials. The author uses the variations in the zero-crossing rate as well as the energy contour for features, a multivariate Gaussian classifier is then used to discriminate between speech and music. In [136], Scheirer and Slaney combine 13 different low-level descriptors for audio representation and investigate the performance of different classifiers (Gaussian mixture model, k-d trees spatial partitioning, and KNNs) for music and speech classification. While the choice of classifier has limited impact on the performance, they show that feature selection is important with the LLDs variance being a better choice than the LLDs themselves.

As audio datasets grew larger, deep learning architectures became applicable to the task of sound recognition. With more than 2 million human-annotated audio clips extracted from YouTube Videos, AudioSet [51] is one such dataset. The ontology features 527 audio classes that range from generic noise categories, e.g. vibrations, to the different voice levels, e.g. whispering, and even includes dedicated categories for animal sounds like rodents noises. The VGG-Sound dataset [30] is also composed of YouTube audio-visual segments, but contrary to AudioSet, VGG-Sound ensures audio-visual correspondence such that the source of sound is visually evident. To leverage such large scale datasets, researchers have adapted the successful computer vision CNN architectures for audio processing. Mel filterbanks spectrograms being equivalent to 2D images, they can be provided directly as input to CNNs to obtain state-of-the-art results on audio classification [66]. Arandjelovic and Zisserman [10] leverage unlabeled video data to train in parallel an audio model and an image model. Provided with the output of each model, a classifier is tasked with predicting if the image and the audio are from the same video or not. More recently, the transformer architecture have shown to improve audio classification performance [112]. Chen et al. [29] also leverage this architecture to tackle the task of audio-visual synchronisation in videos. We explain the transformer architecture in detail in section 2.3.2.

2.2.3 Speech recognition

Spoken language is the most ancient medium for human communication. While modern telecommunications have eased written language communication, speech conversation remains faster and more natural.

Automatic speech recognition (ASR) deals with the automatic extraction of language from the audio signal. Because of all its possible applications, e.g. transcription, dictation, virtual assistants, etc., ASR has attracted considerable interest in recent years.

Traditional approaches to ASR [73, 127] were phoneme-based and required components for the pronunciation, acoustic, and language model to be trained separately. The pronunciation model, associating words with their corresponding sequence of phonemes was manually designed by linguists. The most popular choices for acoustic modeling were hidden markov model while language models were usually n-grams models.

Modern approaches instead tackle ASR end-to-end, using Mel filterbank spectrograms as input and predicting symbols (characters or words) sequences directly. Recurrent networks (RNN), particularly their long short-term memory (LSTM) [67] variant have been the most popular models for that task. The transformer architecture [153] has recently replaced RNN networks thanks to its parallelization and long term dependency processing. Transformers have been augmented with convolutions to model both local and global dependencies more efficiently [60].

In order to train these models, two training approaches dominate the field: CTC and Seq2Seq. The Connectionist Temporal Classification approach (CTC) [57] was the first attempt at end-to-end ASR. The main contribution of CTC is to remove the need for pre-segmented transcript-aligned training data. It is done by interpreting the model outputs as a probability distribution over all possible label sequences alignments. CTC allows learning pronunciation and acoustic modeling end-to-end within a single network. It is however incapable of learning language modeling directly as the model outputs are conditionally independent. Graves et al. [58] revisit CTC and make the network deeper by stacking several bi-directional LSTM layers. A language model is used to decode the model outputs by adapting the transducer architecture [56].

A more recent approach to end-to-end ASR is the sequence-to-sequence model (Seq2Seq) [33, 149]. Contrary to CTC, Seq2Seq does not explicitly align audio and transcript but instead uses a decoder to auto-regressively generate symbols one by one from the encoder outputs, therefore integrating language modeling. The model is trained using cross-entropy loss on each generated symbol. The Seq2Seq approach has been applied to ASR in Listen, Attend and Spell (LAS) [27] and later been derived multiple times [150, 160].

More recently, researchers have worked to leverage the large amount of unannotated speech data available on the internet to pretrain their models in a self-supervised fashion. The wav2vec pretraining [14, 137] trains a model to learn the structure of speech from raw audio. Similar to the Bidirectional Encoder Representations from Transformers (BERT) [40], wav2vec is trained to recover masked speech units by exploiting their

audio context. When finetuned on the ASR task, a wav2vec pretrained model manages to reach better performance than fully-supervised training with 100 times less labeled data.

Another attempt to improve ASR by leveraging unlabeled speech data is semi-supervised learning. It consists in iteratively repeating two steps. In the first step, a pretrained ASR model is used to predict the transcripts of a large-scale raw-audio dataset. In the second step, this automatically annotated dataset is used along with some manually labeled data to retrain the ASR model. By repeating these two steps, both the ASR model and the automatic annotations will progressively get better, each one improving the other. This semi-supervised approach was exploited in [150] and combined with self-supervised pretraining in [160], leading to a word-error-rate as low as 1.5% on the Librispeech dataset [117].

2.3 Natural language processing

Text, either extracted from speech or produced directly in written form, is made of a sequence of symbols. It has historically been the media of choice to store information, which explains why so much textual data is available. Compared to vision or audio which are natural and continuous signals, text is artificial and discrete.

2.3.1 Language representation

Text typically consists of a long sequence of characters or graphemes (e.g., letters of the alphabet). The first step to obtain a representation from text is to break it down into smaller pieces, i.e. tokens. Word-level tokenization has traditionally been employed to decompose text into semantic units (words). However, because the number of words in a language is very large (potentially infinite), it suffers from a major drawback: out-of-vocabulary words. Subword-level tokenization [82, 138, 139, 155] has been proposed as a middle ground between word-level and character-level tokenization. Similarly to character-level tokenization, text can be decomposed into a finite set of tokens without out-of-vocabulary words. Similarly to word-level tokenization, the tokens obtained with subword-level tokenization carry semantics and are close to morphemes (smallest meaningful lexical item in a language). Once tokenized, a piece of text becomes a sequence of tokens that can be used to extract text-level or token-level representations (Fig. 2.5).

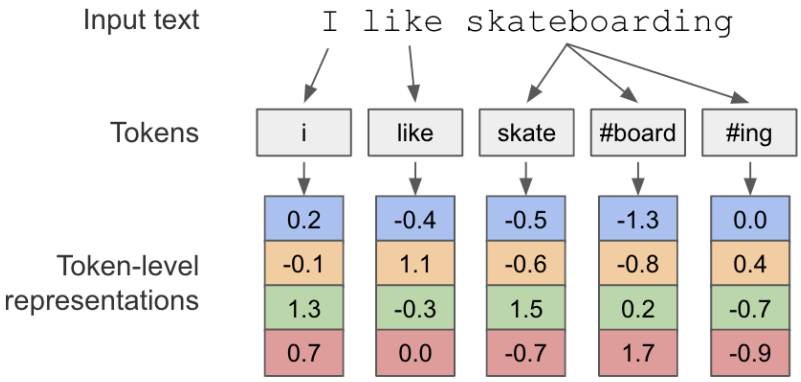

Figure 2.5: Token-level representations of an input text through tokenization. Subword-level tokenization allows the decomposition of words into semantic pieces while avoiding out-of-vocabulary issues. A lookup table can then be used to associate each token of the vocabulary to a vector representation.

One of the earliest and simplest ways to obtain a representation from a textpiece is Bag of Words (BOW) [97, 168]. It consists in counting the occurrence frequency of each token in the text to obtain an histogram. That histogram can then be used as a text-level representation to calculate similarities between documents or as input to a classifier. The simplicity of BOW is counterbalanced by several drawbacks, mainly sparsity and insensitivity to word order.

Word embeddings have been developed to obtain text representations at the word level instead of at the text level. Word embeddings associate each word with a vector that encodes its semantics such that relative similarity between embeddings in vector space correlate with words semantic similarity (Fig. 2.6). An unsupervised approach to learn word embeddings from unlabeled text was introduced by Mikolov et al. in 2013 with Word2Vec [105]. Leveraging the skip-gram architecture, Word2Vec embeddings are randomly initialized and iteratively optimized using the representation of a given word to predict its context words.

Using word embeddings, a text can then be transformed to a sequence of vectors to be processed by a language model to obtain a text-level representation. Word embeddings have been used both directly as input features to a language model (frozen features) or as initialization of its first layer lookup table (finetuned features). Prior to 2017, the model of choice

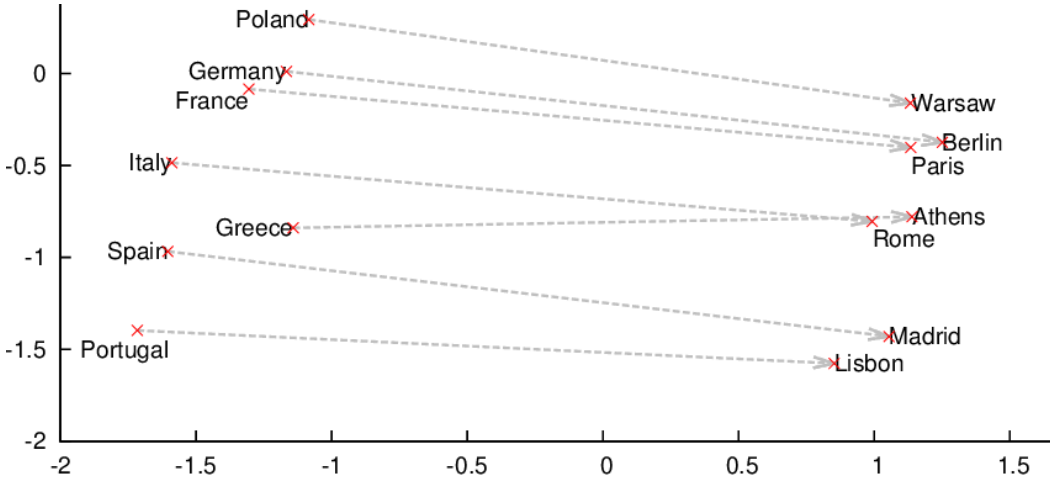

Figure 2.6: Two-dimensional PCA projection of word2vec embeddings of countries and their capital cities. This figure illustrates how the word2vec model managed to position words in vector space in a way that concepts such as 'city = capital(country)' approximately correspond to a vector addition. (Image courtesy of Mikolov et al. [106])

for language modeling were recurrent neural networks. RNNs sequentially process variable length sequences of inputs and use an internal memory to store information from previous inputs. Because of their recurrent architecture, RNNs however suffer from several drawbacks such as sequential processing and limited contextual range. The introduction of long short-term memory (LSTM) [67] allows RNNs to learn longer term dependencies and reduces the vanishing gradient problem. The sequential computation of RNNs however remains an issue for parallelization, which is required for training on long sentences. The introduction of the Transformer architecture solved this parallelization issue, as we will explain in the next section.

2.3.2 Transformers

Introduced in 2017, the transformer [153] is a recurrence-free architecture capable of processing long sequences of inputs in parallel. It is also convolution-free and leverages the self-attention mechanism to let any input attend to all the others, independently of their position in the sequence. The transformer actually processes the input sequence as a set of embeddings, without ordering, it is therefore desirable to provide information about the relative position of the vectors in the sequence. This

position information is additively encoded in the input sequence using either sinusoidal encodings or learned positional embeddings.

The transformer encoder consists in a stack of N identical transformer layers (Fig. 2.7, right), each composed of two sublayers: a multi-head self-attention layer (MHA) and a fully connected feed-forward network (FFN). The transformation of a sequence X by a transformer layer TL is given by:

$$TL(X) = f(g(X)) \tag{2.1}$$
$$\text{with } f(X) = LN(X + FFN(X))$$
$$\text{and } g(X) = LN(X + MHA(X))$$

where TL is the transformer layer transformation, $X \in \mathbb{R}^{n \times d_{model}}$ is the input sequence of n embeddings, LN is layer normalization [13], FFN is a fully connected feed-forward network and MHA is a multi-head self-attention layer which will be detailed in the following paragraph.

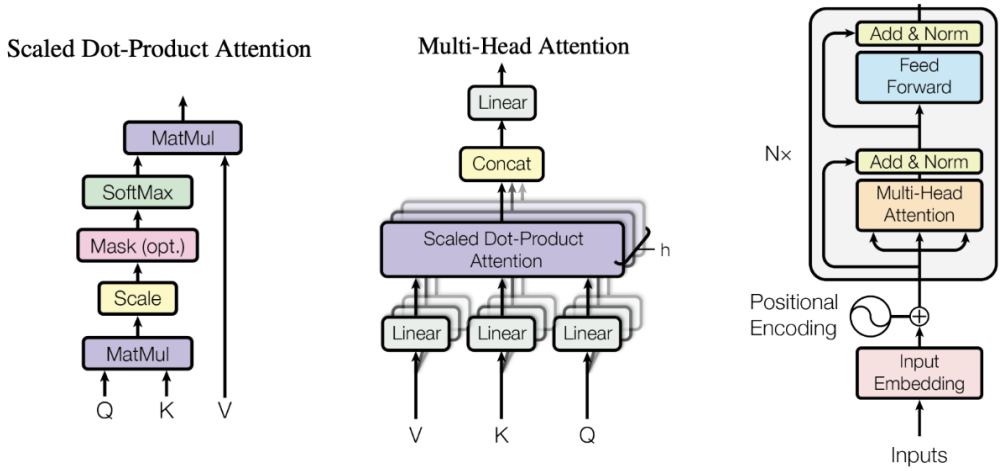

Figure 2.7: The scaled dot-product attention mechanism employed in the transformer lets each input embedding attend to all the others (left). Attention is performed in parallel over multiple heads to increase the width the network (middle). The inputs are embedded, position-encoded and then processed by the N layers of the transformer encoder (right). (Image courtesy of Vaswani et al. [153])

The attention mechanism processes each embedding in the input sequence by first linearly transforming it into three vectors: a query, a key and a value. Similarities are then computed between the query and all

keys using the dot product. This lets each embedding attend to all other embeddings in the sequence and find which ones are most semantically relevant to the query. Finally, a softmax operation is applied to the similarities to obtain attention weights and the result is computed as the weighted sum of all values (Fig. 2.7, left).

In practice, the attention function is computed in vectorized form:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{T})V \tag{2.2}$$

Where Q, K and V are the matrices of queries, keys and values of the input embeddings, and T the softmax temperature.

To increase the width of the model, attention is performed in parallel over multiple attention heads. Each attention head transforms the input X into queries, keys and values using its own projection matrices W_i^Q, W_i^K and W_i^V. The outputs from the different heads are concatenated and linearly projected with the W^O matrix to d_{model} dimension as shown is figure 2.7 (middle):

$$\begin{aligned} \text{MHA}(X) &= \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O \\ \text{where head}_\text{i} &= \text{Attention}(XW_i^Q, XW_i^K, XW_i^V) \end{aligned}$$

At the output of the transformer encoder, we obtain the same number of embeddings and with the same dimension as in the input, but these have been contextualized. By letting each embedding attend to all the others, the transformer has allowed information to flow between the original vectors of the sequence. In addition to text, the transformer has also become a popular model for vision and audio processing. In the following section, we will see how to train such a model to contextualize its inputs.

2.3.3 Masked language modeling

Similar to computer vision and audio processing, natural language processing has been restrained by the limited availability of large annotated datasets. Annotated text datasets are usually small scale and target a single task, e.g. named entity recognition, text classification, sentiment analysis.

BERT pretraining [40] is designed to leverage the large amount of unannotated textual data available on the internet for language understanding. Its main contribution is the introduction of masked language modeling (MLM) to pretrain a transformer encoder in a self-supervised fashion. MLM consists in randomly corrupting some words in the input sentence and training a model to recover the original words (Fig. 2.8).

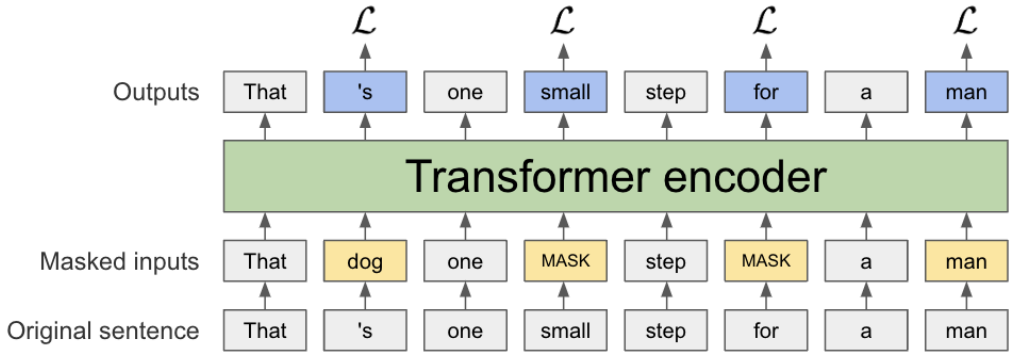

Figure 2.8: Masked language modeling. Randomly selected tokens (yellow) are either masked, kept the same, or replaced by a random token. Cross-entropy loss is applied to the corresponding output embeddings (blue) after linear transformation and softmax.

The MLM objective forces the model to learn some level of language understanding to find out what were the masked tokens. It also learns to contextualize each of the input tokens to determine whether a token is correct or if it has been randomly replaced. MLM pretraining obtains large gains on a variety of NLP tasks after finetuning of the model and has been adapted and extended to computer vision [18] and audio processing [14].

With the notable introduction of Transformer and MLM, we have observed in recent years a convergence of architectures and training strategies to process the different modalities that are language, audio and vision. In the next chapters, we build on these techniques and go one step further to target the extraction of cross-modal semantics by jointly processing the different video modalities.

Chapter 3

Fusing video modalities

Video is one of the most popular forms of media due to its ability to capture dynamic events and its natural appeal to our visual and auditory senses. Online video platforms are playing a major role in promoting this form of media. However, the billions of hours of video available on such platforms are unusable if we cannot access them effectively, for example, by retrieving relevant content through queries.

In this chapter, we tackle the tasks of caption-to-video and video-to-caption retrieval. In the first task of caption-to-video retrieval, we are given a query in the form of a caption (e.g., “How to build a house") and the goal is to retrieve the videos best described by it (i.e., videos explaining how to build a house). In practice, given a test set of caption-video pairs, our aim is to provide, for each caption query, a ranking of all the video candidates such that the video associated with the caption query is ranked as high as possible. On the other hand, the task of video-to-caption retrieval focuses on finding among a collection of caption candidates the ones that best describe the query video.

A common approach for the retrieval problem is similarity learning [157], where we learn a function of two elements (a query and a candidate) that best describes their similarity. All the candidates can then be ranked according to their similarity with the query. In order to perform this ranking, the captions as well as the videos are represented in a common multi-dimensional embedding space, wherein similarities can be computed as a dot product of their corresponding representations. The critical question here is how to learn accurate representations of both caption and video to base our similarity estimation on.

The problem of learning representation of text has been extensively studied, leading to various methods [40, 67, 105, 153, 168], which can be used to encode text captions. In contrast to these advances, learning effective video representation continues to be a challenge, and forms the focus of our work. This is in part due to the multimodal and temporal nature of

Figure 3.1: When matching a text query with videos, the inherent cross-modal and temporal information in videos needs to be leveraged effectively, for example, with a video encoder that handles all the constituent modalities (appearance, audio, speech) jointly across the entire duration of the video. In this example, a video encoder will be able to distinguish between "someone walking *to*" and "someone walking *away*" only if it exploits the temporal information of events occurring in the video (red arrows). Also, in order to understand that a "motorbike failed to start", it needs to use cross-modal information (e.g., absence of noise after someone tried to start the engine, orange arrow).

video. Video data not only varies in terms of appearance, but also in possible motion, audio, overlaid text, speech, etc. Leveraging cross-modal relations thus forms a key to building effective video representations. As illustrated in Fig. 3.1, cues jointly extracted from all the constituent modalities are more informative than handling each modality independently. Hearing a motor sound right after seeing someone starting a bike tells us that the running bike is the visible one and not a background one. Another example is the case of a video of "a crowd listening to a talk", neither of the modalities "appearance" or "audio" can fully describe the scene, but when processed together, higher level semantics can be obtained.

Recent work on video retrieval does not fully exploit such cross-modal high-level semantics. They either ignore the multi-modal signal [102], treat modalities separately [103], or only use a gating mechanism to modulate certain modality dimensions [90]. Another challenge in representing video is its temporality. Due to the difficulty in handling variable duration of

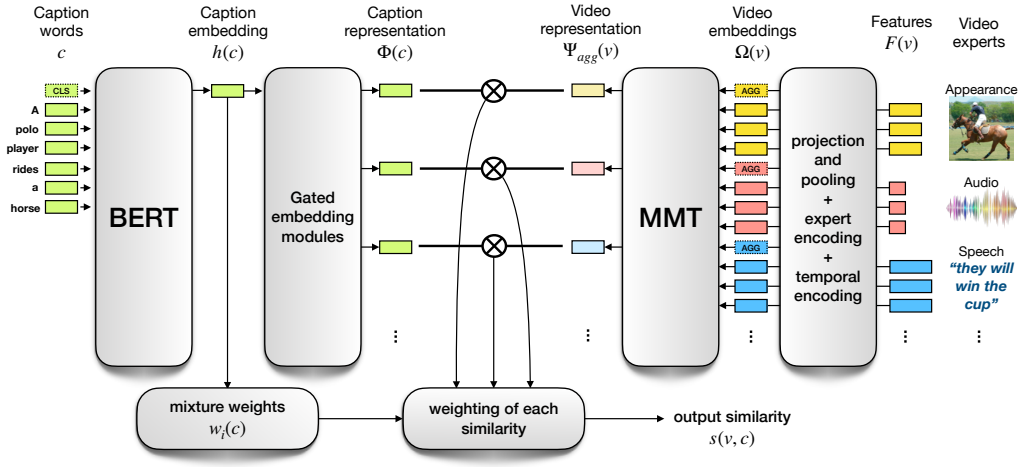

Figure 3.2: Our cross-modal framework for similarity estimation. We use our Multi-modal Transformer (MMT, right) to encode video, and BERT (left) for text.

videos, current approaches [90, 103] discard long-term temporal information by aggregating descriptors extracted at different moments in the video. We argue that this temporal information can be important to the task of video retrieval. As shown in Fig. 3.1, a video of "someone walking *to* an object" and "someone walking *away* from an object" will have the same representation once pooled temporally, however, the movement of the person relative to the object is potentially important in the query.

We address the temporal and multi-modal challenges posed in video data by introducing our multi-modal transformer. It performs the task of processing features extracted from different modalities at different moments in video and aggregates them in a compact representation. Building on the transformer architecture [153], our multi-modal transformer exploits the self-attention mechanism to gather valuable cross-modal and temporal cues about events occurring in a video. We integrate our multi-modal transformer in a cross-modal framework, as illustrated in Fig. 3.2, which leverages both captions and videos, and estimates their similarity.

Contributions. In this chapter, we make the following three contributions: (i) First, we introduce a novel video encoder architecture for retrieval: Our multi-modal transformer processes effectively multiple modality features extracted at different times. (ii) We thoroughly investigate different architectures for language embedding, and show the

superiority of the BERT model for the task of video retrieval. (iii) By leveraging our novel cross-modal framework we outperform prior state of the art for the task of video retrieval on MSRVTT [159], ActivityNet [80] and LSMDC [130] datasets. Our approach is also the winning solution in the CVPR 2020 Video Pentathlon Challenge [49].

3.1 Related work

In this section we present previous work on visual-language retrieval as well as on fusion of pre-computed video features.

Visual-language retrieval. Harwath et al. [62] perform image and audio-caption retrieval by embedding audio segments and image regions in the same space and requiring high similarity between each audio segment and its corresponding image region. The method presented in [85] takes a similar approach for image-text retrieval by embedding images regions and words in a joint space. A high similarity is obtained for images that have matching words and image regions. Burns et al. [22] perform an analysis of the different word embeddings and language models (Word2Vec [105], LSTM [67], BERT [40], etc.) used for image-text retrieval and other vision-language tasks. They show that the pretrained and frozen BERT model performs relatively poorly compared to a LSTM or even a simpler average embedding model. In this work, we find that a pretrained BERT outperforms other language models, but it needs to be finetuned.

For videos, JSFusion [162] estimates video-caption similarity through dense pairwise comparisons between each word of the caption and each frame of the video. In this work, we instead estimate both a video embedding and a caption embedding and then compute the similarity between them. In [54], the authors propose a two-branches architecture that models the interactions between different levels of granularity in both the visual modality and the text modality. Zhang et al. [166] perform paragraph-to-video retrieval by assuming a hierarchical decomposition of the video and paragraph. Our method does not assume that the video can be decomposed into clips that align with sentences of the caption. A recent alternative is creating separate embedding spaces for different parts of speech (e.g., noun or verb) [154]. In contrast to this method, we do not pre-process the sentences but encode them directly through BERT. Another work [104] leverages the large number of instructional videos in the HowTo100M dataset, but does not fully exploit the temporal relations. The approach presented in this chapter instead relies on longer

segments extracted from HowTo100M videos in order to learn temporal dependencies and address the problem of misalignment between speech and visual features.

Fusion of pre-computed video features. Because of the small scale of manually annotated text-to-video retrieval datasets as well as the high computational cost of processing pixels and raw audio signal directly, a popular approach for video retrieval has been to use pre-computed features from 'expert' models. These models are trained for diverse tasks and on multiple modalities such as face [120], scene [170] and object recognition, action classification [26] and sound classification [51].

Mithun et al. [107, 108] use three experts (Object, Activity and Place) to compute three corresponding text-video similarities. These experts however do not collaborate together as their respective similarities are simply summed together. A related approach [103] uses pre-computed features from experts for text to video retrieval, where the overall similarity is obtained as a weighted sum of each expert's similarity. A recent extension [90] to this mixture of experts model uses a collaborative gating mechanism for modulating each expert feature according to the other experts. However, this collaborative gating mechanism only strengthens (or weakens) some dimensions of the input signal in a single step, and is therefore not able to capture high level inter-modality information. Our multi-modal transformer overcomes this limitation by attending to all available modalities over multiple self-attention layers. More recently, several works [43, 125] have shown the superiority of using the CLIP [128] model to extract appearance features, therefore leveraging the 400 million (image, caption) pairs it was trained on.

3.2 Methodology

Our overall method relies on learning a function s to compute the similarity between two elements: text and video, as shown in Fig. 3.2. We then rank all the videos (or captions) in the dataset, according to their similarity with the query caption (or video) in the case of text-to-video (or video-to-text) retrieval. In other words, given a dataset of n video-caption pairs $\{(v_1, c_1), ..., (v_n, c_n)\}$, the goal of the learnt similarity function $s(v_i, c_j)$, between video v_i and caption c_j, is to provide a high value if $i = j$, and a low one if $i \neq j$. Estimating this similarity (described in Section 3.2.3) requires accurate representations for the video as well as the caption. Fig. 3.2 shows

the two parts focused on producing these representations (presented in Sections 3.2.1 and 3.2.2 respectively) in our cross-modal framework.

3.2.1 Video representation

The video-level representation is computed by our proposed multi-modal transformer (MMT). MMT follows the architecture of the transformer encoder presented in [153]. It consists of stacked self-attention layers and fully collected layers. MMT's input $\Omega(v)$ is a set of embeddings, all of the same dimension d_{model}. Each of them embeds the semantics of a feature, its modality, and the time in the video when the feature was extracted. This input is given by:

$$\Omega(v) = F(v) + E(v) + T(v), \tag{3.1}$$

In the following, we describe those three components.

Features F**.** In order to learn an effective representation from different modalities inherent in video data, we begin with video feature extractors called "experts" [90, 103, 107, 162]. In contrast to previous methods, we learn a joint representation leveraging both cross-modal and long-term temporal relationships among the experts. We use N pretrained experts $\{F^n\}_{n=1}^N$. Each expert is a model trained for a particular task that is then used to extract features from video. For a video v, each expert extracts a sequence $F^n(v) = [F_1^n, ..., F_K^n]$ of K features.

The features extracted by our experts encode the semantics of the video. Each expert F^n outputs features in $\mathbb{R}^{d_n}$. In order to project the different expert features into a common dimension d_{model}, we learn N linear layers (one per expert) to project all the features into $\mathbb{R}^{d_{model}}$.

A transformer encoder produces an embedding for each of its feature inputs, resulting in several embeddings for an expert. In order to obtain a unique embedding for each expert, we define an aggregated embedding F_{agg}^n that will collect and contextualize the expert's information. We initialize this embedding with a max pooling aggregation of all the corresponding expert's features as $F_{agg}^n = maxpool(\{F_k^n\}_{k=1}^K)$. The sequence of input features to our video encoder then takes the form:

$$F(v) = [F_{agg}^1, F_1^1, ..., F_K^1, ..., F_{agg}^N, F_1^N, ..., F_K^N]. \tag{3.2}$$

Expert embeddings E**.** In order to process cross-modality information, our MMT needs to identify which expert it is attending to. We learn N embeddings $\{E_1, ..., E_N\}$ of dimension d_{model} to distinguish between embeddings of different experts. Thus, the sequence of expert embeddings to our video

encoder takes the form:

$$E(v) = [E^1, E^1, ..., E^1, ..., E^N, E^N, ..., E^N]. \tag{3.3}$$

Temporal embeddings T**.** They provide temporal information about the time in the video where each feature was extracted to our multi-modal transformer. Considering videos of a maximum duration of t_{max} seconds, we learn $D = |t_{max}|$ embeddings $\{T_1, ..., T_D\}$ of dimension d_{model}. Each expert feature that has been extracted in the time range $[t, t+1)$ will be temporally embedded with T_{t+1}. For example, a feature extracted at 7.4s in the video will be temporally encoded with temporal embedding T_8. We learn two additional temporal embeddings T_{agg} and T_{unk}, which encode aggregated features and unknown temporal information features (for experts whose temporal information is unknown), respectively. The sequence of temporal embeddings of our video encoder then takes the form:

$$T(v) = [T_{agg}, T_1, ..., T_D, ..., T_{agg}, T_1, ..., T_D]. \tag{3.4}$$

Multi-modal Transformer. The video embeddings $\Omega(v)$ defined as the sum of features, expert and temporal embeddings in (3.1), as shown in Fig. 3.3, are input to the transformer. They are given by: $\Omega(v) = F(v) + E(v) + T(v) = [\omega^1_{agg}, \omega^1_1, ..., \omega^1_K, ..., \omega^N_{agg}, \omega^N_1, ..., \omega^N_K]$. MMT contextualises its input $\Omega(v)$ and produces the video representation $\Psi_{agg}(v)$. As illustrated in Fig. 3.2, we only keep the aggregated embedding per expert. Thus, our video representation $\Psi_{agg}(v)$ consists of the output embeddings corresponding to the aggregated features, i.e.,

$$\Psi_{agg}(v) = MMT(\Omega(v)) = [\psi^1_{agg}, ..., \psi^N_{agg}]. \tag{3.5}$$

The advantage of our MMT over the state-of-the-art collaborative gating mechanism [90] is two-fold: First, the input embeddings are not simply modulated in a single step but iteratively refined through several layers featuring multiple attention heads. Second, we do not limit our video encoder with a temporally aggregated feature for each expert, but provide all the extracted features instead, along with a temporal encoding describing at what moment of the video they were extracted from. Thanks to its self-attention modules, each layer of our multi-modal transformer is able to attend to all its input embeddings, thus extracting semantics of events occurring in the video over several modalities.

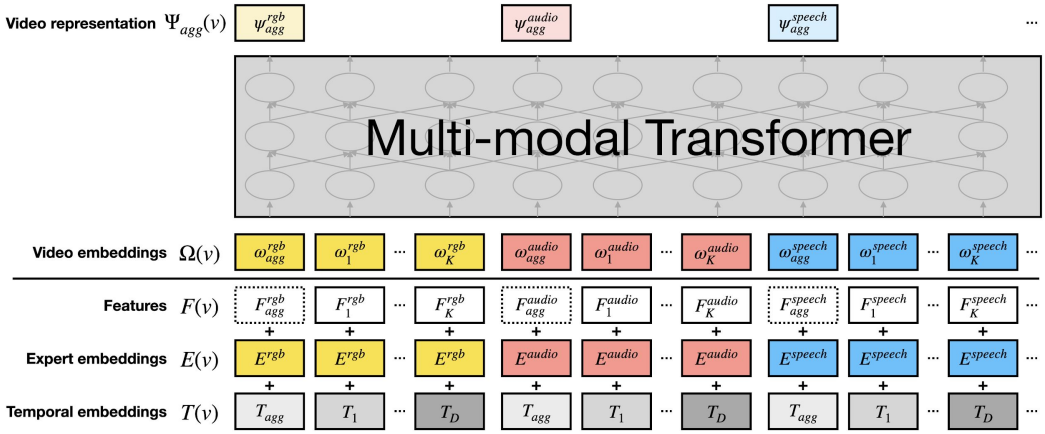

Figure 3.3: Multi-modal transformer inputs and outputs. We combine feature semantics F, expert information E, and temporal cues T to form our video embeddings $\Omega(v)$, which are input to MMT to obtain the video representation $\Psi_{agg}(v)$.

3.2.2 Caption representation

We compute our caption representation $\Phi(c)$ in two stages: first, we obtain an embedding $h(c)$ of the caption, and then project it with a function g into N different spaces as $\Phi = g \circ h$. For the embedding function h, we use a pretrained BERT model [40]. Specifically, we extract our single caption embedding $h(c)$ from the [CLS] output of BERT. In order to match the size of this caption representation with that of video, we learn for function g as many gated embedding modules [103] as there are video experts. Our caption representation then consists of N embeddings, represented by $\Phi(c) = \{\phi^i\}_{i=1}^N$.

3.2.3 Similarity estimation

We compute our final video-caption similarity s, as a weighted sum of each expert i's video-caption similarity $\langle \phi^i, \psi^i_{agg} \rangle$. It is given by:

$$s(v,c) = \sum_{i=1}^{N} w_i(c) \langle \phi^i, \psi^i_{agg} \rangle, \tag{3.6}$$

where $w_i(c)$ represents the weight for the ith expert. To obtain these mixture weights, we follow [103] and process our caption representation $h(c)$

through a linear layer and then perform a softmax operation, i.e.,

$$w_i(c) = \frac{e^{h(c)^\top a_i}}{\sum_{j=1}^{N} e^{h(c)^\top a_j}}, \tag{3.7}$$

where $(a_1, ..., a_N)$ are the weights of the linear layer. The intuition behind using a weighted sum is that a caption may not describe all the inherent modalities in video uniformly. For example, in the case of a video with a person in a red dress singing opera, the caption "a person in a red dress" provides no information relevant for audio. On the contrary, the caption "someone is singing" should focus on the audio modality for computing similarity. Note that $w_i(c), \phi^i$ and ψ^i_{agg} can all be precomputed offline for each caption and for each video, and therefore the retrieval operation only involves dot product operations.

3.2.4 Training

We train our model with the bi-directional max-margin ranking loss [75]:

$$\mathcal{L} = \frac{1}{B} \sum_{i=1}^{B} \sum_{j \neq i} \Big[\max(0, s_{ij} - s_{ii} + m) + \max(0, s_{ji} - s_{ii} + m) \Big], \tag{3.8}$$

where B is the batch size, $s_{ij} = s(v_i, c_j)$, the similarity score between video v_i and caption c_j, and m is the margin. This loss enforces the similarity for true video-caption pairs s_{ii} to be higher than the similarity of negative samples s_{ij} or s_{ji}, for all $i \neq j$, by at least m.

3.3 Experiments

3.3.1 Datasets and Metrics

HowTo100M [104]. It is composed of more than 1 million YouTube instructional videos, along with automatically-extracted speech transcriptions, which form the captions. These captions are naturally noisy, and often do not describe the visual content accurately or are temporally misaligned with it. We use this dataset only for pretraining.
MSRVTT [159]. This dataset is composed of 10K YouTube videos, collected using 257 queries from a commercial video search engine. Each video is 10 to 30s long, and is paired with 20 natural sentences describing it, obtained

from Amazon Mechanical Turk workers. We use this dataset for training from scratch and also for finetuning. We report results on the train/test splits introduced in [162] that uses 9000 videos for training and 1000 for test. We refer to this split as "1k-A". We also report results on the train/test split in [103] that we refer to as "1k-B". Unless otherwise specified, our MSRVTT results are with "1k-A".
ActivityNet Captions [80]. It consists of 20K YouTube videos temporally annotated with sentence descriptions. We follow the approach of [166], where all the descriptions of a video are concatenated to form a paragraph. The training set has 10009 videos. We evaluate our video-paragraph retrieval on the "val1" split (4917 videos). We use ActivityNet for training from scratch and finetuning.
LSMDC [130]. It contains 118,081 short video clips ($\sim$ 4–5s) extracted from 202 movies. Each clip is annotated with a caption, extracted from either the movie script or the audio description. The test set is composed of 1000 videos, from movies not present in the training set. We use LSMDC for training from scratch and also finetuning.

Metrics. We evaluate the performance of our model with standard retrieval metrics: recall at rank N (R@N, higher is better), median rank (MdR, lower is better) and mean rank (MnR, lower is better). For each metric, we report the mean and the standard deviation over experiments with 3 random seeds.

3.3.2 Implementation details

Pretrained experts. Recall that our video encoder uses pretrained experts models for extracting features from each video modality. We use the following seven experts.
Motion features are extracted from S3D [156] trained on the Kinetics action recognition dataset.
Audio features are extracted using VGGish model [66] trained on YT8M.
Scene embeddings are extracted from DenseNet-161 [69] trained for image classification on the Places365 dataset [170].
OCR features are obtained in three stages. Overlaid text is first detected using the pixel link text detection model. The detected boxes are then passed through a text recognition model trained on the Synth90K dataset. Finally, each character sequence is encoded with word2vec [105] embeddings.
Face features are extracted in two stages. An SSD face detector is used to extract bounding boxes, which are then passed through a ResNet50 trained

for face classification on the VGGFace2 dataset.
Speech transcripts are extracted using the Google Cloud Speech to Text API, with the language set to English. The detected words are then encoded with word2vec.
Appearance features are extracted from the final global average pooling layer of SENet-154 [68] trained for classification on ImageNet.

For scene, OCR, face, speech and appearance, we use the features publicly released by [90], and compute the other features ourselves.

Training. For each dataset, we run a grid search on the corresponding validation set to estimate the hyperparameters. We use the Adam optimizer for all our experiments, and set the margin of the bidirectional max-margin ranking loss to 0.05. We also freeze our pretrained expert models.

When pretraining on HowTo100M, we use a batch size of 64 video-caption pairs, an initial learning rate of 5e-5, which we decay by a multiplicative factor 0.98 every 10K optimisation steps, and train for 2 million steps. Given the long duration of most of the HowTo100M videos, we randomly sample 100 consecutive words in the caption, and keep 100 consecutive seconds of video data, closest in time to the selected words.

When training from scratch or finetuning on MSRVTT or LSMDC, we use a batch size of 32 video-caption pairs, an initial learning rate of 5e-5, which we decay by a multiplicative factor 0.95 every 1K optimisation steps. We train for 50K steps. We use the same settings when training from scratch or finetuning on ActivityNet, except for 0.90 as the multiplicative factor.

To compute our caption representation $h(c)$, we use the "BERT-base-cased" checkpoint of the BERT model and finetune it with a dropout probability of 10%. To compute our video representation $\Psi_{agg}(v)$, we use MMT with 4 layers and 4 attention heads, a dropout probability of 10%, a hidden size d_{model} of 512, and an intermediate size of 3072.

For datasets with short videos (MSRVTT and LSMDC), we use all the 7 experts and limit video input to 30 features per expert, and BERT input to the first 30 wordpieces. For datasets containing longer videos (HowTo100M and ActivityNet), we only use motion and audio experts, and limit our video input to 100 features per expert and our BERT input to the first 100 wordpieces. In cases where an expert is unavailable for a given video, e.g., no speech was detected, we set the aggregated feature F^n_{agg} to a zero vector. We refer the reader to the supplementary material for a study of the model complexity.

3.3.3 Ablation studies and comparisons

We will first show the advantage of pretraining our model on a large-scale, uncurated dataset. We will then perform ablations on the architecture used for our language and video encoders. Finally, we will present the relative importance of the pretrained experts used in this work, and compare with related methods.

Pretraining. Table 3.1 shows the advantage of pretraining on HowTo100M, before finetuning on the target dataset (MSRVTT in this case). We also evaluated the impact of pretraining before finetuning on ActivityNet and LSMDC; see Table 3.5 and Table 3.6.

Table 3.1: Advantage of pretraining on HowTo100M then finetuning on MSRVTT compared to training from scratch on MSRVTT or pretraining on HowTo100M only (zero shot). Impact of removing the stop words. Performance reported on MSRVTT.

		Text $\longrightarrow$ *Video*		
Method	Caption	R@5↑	MdR↓	MnR↓
Pretraining w/o finetuning	all words	6.9	160.0	240.2
	w/o stop words	**14.4**	**66.0**	**148.1**
Training from scratch	all words	**54.0**$_{\pm 0.2}$	**4.0**$_{\pm 0.0}$	**26.7**$_{\pm 0.9}$
	w/o stop words	50.0$_{\pm 0.6}$	5.3$_{\pm 0.5}$	28.5$_{\pm 0.9}$
Pretraining then finetuning	all words	**57.1**$_{\pm 1.0}$	**4.0**$_{\pm 0.0}$	**24.0**$_{\pm 0.8}$
	w/o stop words	55.0$_{\pm 0.7}$	4.3$_{\pm 0.5}$	24.3$_{\pm 0.3}$

Language encoder. We evaluated several architectures for caption representation, as shown in Table 3.2. Similar to the observation made in [22], we obtain poor results from a frozen, pretrained BERT. Using the [CLS] output from a pretrained and frozen BERT model is in fact the worst result. We suppose this is because the output was not trained for caption representation, but for a very different task: next sentence prediction. Finetuning BERT greatly improves performance; it is the best result. We also compare with GrOVLE [22] embeddings, frozen or finetuned, aggregated with a max-pooling operation or a 1-layer LSTM and a fully-connected layer. We show that pretrained BERT embeddings aggregated by a max-pooling operation perform better than GrOVLE embeddings processed by a LSTM (best results from [22] for the text-to-clip task).

Table 3.2: Comparison of different architectures for caption embedding when training from scratch on MSRVTT.

Word embeddings		Aggregation	*Text* ⟶ *Video* R@5↑	MdR↓	MnR↓
GrOVLE	frozen	maxpool	$31.8_{\pm 0.4}$	$14.7_{\pm 0.5}$	$63.1_{\pm 1.3}$
		LSTM	$36.4_{\pm 0.8}$	$10.3_{\pm 0.9}$	$44.2_{\pm 0.1}$
	finetuned	maxpool	$34.6_{\pm 0.1}$	$12.0_{\pm 0.0}$	$52.3_{\pm 0.8}$
		LSTM	$40.3_{\pm 0.5}$	$8.7_{\pm 0.5}$	$38.1_{\pm 0.7}$
BERT	frozen	maxpool	$39.4_{\pm 0.8}$	$9.7_{\pm 0.5}$	$46.5_{\pm 0.2}$
		LSTM	$36.4_{\pm 1.8}$	$10.7_{\pm 0.5}$	$42.2_{\pm 0.6}$
	finetuned	maxpool	$44.2_{\pm 1.2}$	$7.3_{\pm 0.5}$	$35.6_{\pm 0.4}$
		LSTM	$40.1_{\pm 1.0}$	$8.7_{\pm 0.5}$	$37.4_{\pm 0.5}$
	frozen	BERT-frozen	$17.1_{\pm 0.2}$	$34.7_{\pm 1.2}$	$98.8_{\pm 0.8}$
	finetuned	BERT-finetuned	$\mathbf{54.0}_{\pm 0.2}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.9}$

We also analysed the impact of removing stop words from the captions in Table 3.1. In a zero-shot setting, i.e., trained on HowTo100M, evaluated on MSRVTT without finetuning, removing the stop words helps generalize, by bridging the domain gap—HowTo100M speech is very different from MSRVTT captions. This approach was adopted in [102]. However, we observe that when finetuning, it is better to keep all the words as they contribute to the semantics of the caption.

Video encoder. We evaluated the influence of different architectures for computing video embeddings on the MSRVTT 1k-A test split.

In Table 3.3a, we evaluate variants of our encoder architecture and its input. Similar to [103], we experiment with directly computing the caption-video similarities on each max-pooled expert features, i.e., no video encoder (NONE in the table). We compare this with the collaborative gating architecture (COLL) [90] and our MMT variant using only the aggregated features as input. For the first two variants without MMT, we adopt the approach of [103] to deal with missing modalities by re-weighting $w_i(c)$. We also show the superior performance of our multi-modal transformer in contextualising the different modality embeddings compared to the collaborative gating approach. We argue that our MMT is able to extract cross-modal information in a multi-stage architecture compared to collaborative gating, which is limited to modulating the input embeddings. Table 3.3a also highlights the advantage of providing

Table 3.3: Ablation studies on the video encoder of our framework with MSRVTT. **(a) Influence of the architecture and input.** With max-pooled features as input, we compare our transformer architecture (MMT) with the variant not using an encoder (NONE) and the one with Collaborative Gating [90] (COLL). We also show that MMT can attend to all extracted features, as detailed in the text. **(b) Importance of initializing F^n_{agg} features.** We compare zero-vector initialisation, mean pooling and max pooling of the expert features. **(c) Influence of the size of the multi-modal transformer.** We compare different values for number-of-layers $\times$ number-of-attention-heads.

(a) Encoder architecture and input

		Text $\longrightarrow$ *Video*		
Encoder	Input	R@5↑	MdR↓	MnR↓
NONE	max pool	$50.9_{\pm 1.5}$	$5.3_{\pm 0.5}$	$28.6_{\pm 0.5}$
COLL	max pool	$51.3_{\pm 0.8}$	$5.0_{\pm 0.0}$	$29.5_{\pm 1.8}$
MMT	max pool	$52.5_{\pm 0.7}$	$5.0_{\pm 0.0}$	$27.2_{\pm 0.7}$
MMT	shuffled feats	$53.3_{\pm 0.2}$	$5.0_{\pm 0.0}$	$27.4_{\pm 0.7}$
MMT	ordered feats	$\mathbf{54.0}_{\pm 0.2}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.9}$

(b) F^n_{agg} initialisation

	Text $\longrightarrow$ *Video*		
F^n_{agg} init	R@5↑	MdR↓	MnR↓
zero	$50.2_{\pm 0.9}$	$5.7_{\pm 0.5}$	$28.5_{\pm 1.3}$
mean pool	$\mathbf{54.2}_{\pm 0.3}$	$5.0_{\pm 0.0}$	$27.1_{\pm 0.9}$
max pool	$54.0_{\pm 0.2}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.9}$

(c) Model size

		Text $\longrightarrow$ *Video*		
Layers	Heads	R@5↑	MdR↓	MnR↓
2	2	$53.2_{\pm 0.4}$	$5.0_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.4}$
4	4	$\mathbf{54.0}_{\pm 0.2}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.9}$
8	8	$53.9_{\pm 0.3}$	$4.7_{\pm 0.5}$	$\mathbf{26.7}_{\pm 0.7}$

MMT with **all** the extracted features, instead of only aggregated ones. Temporally aggregating each expert's features ignores information about multiple events occurring in a same video (see the last three rows). As shown by the influence of ordered and randomly shuffled features on the performance, MMT has the capacity to make sense of the relative ordering of events in a video.

Table 3.3b shows the importance of initialising the expert aggregation feature F^n_{agg}. Since the output of our video encoder is extracted from the "agg" columns, it is important to initialise them with an appropriate representation of the experts' features. The transformer being a residual network architecture, initializing F^n_{agg} input embeddings with a zero vector leads to a low performance. Initializing with max pooling aggregation of each expert performs better than mean pooling. Finally, we analyze the impact of the size of our multi-modal transformer model in Table 3.3c. A model with 4 layers and 4 attention heads outperforms both a smaller model (2 layers and 2 attention heads) and a larger model (8 layers and 8 attention heads).

Comparison of the different experts. In Figure 3.4, we show an ablation study when training our model on MSRVTT using only one expert (left), using all experts but one (middle), or gradually adding experts by greedy search (right). In the case of using only one expert, we note that the motion expert provides the best results. We attribute the poor performance of OCR, speech and face to the fact that they are absent from many videos, thus resulting in a zero vector input to our video encoder. While the scene expert shows a decent performance, if used alone, it does not contribute when used along other experts, perhaps due to the semantics it encodes being captured already by other experts like appearance or motion. On the contrary, the audio expert alone does not provide a good performance, but it contributes the most when used in conjunction with the others, most likely due to the complementary cues it provides, compared to the other experts.

Comparison to prior state of the art. We compare our method on three datasets: MSRVTT (Table 3.4), ActivityNet (Table 3.5) and LSMDC (Table 3.6). While MSRVTT and LSMDC contain short video-caption pairs (average video duration of 13s for MSRVTT, one-sentence captions), ActivityNet contains much longer videos (several minutes) and each video is captioned with multiple sentences. We consider the concatenation of all these sentences as the caption. We show that our method obtains

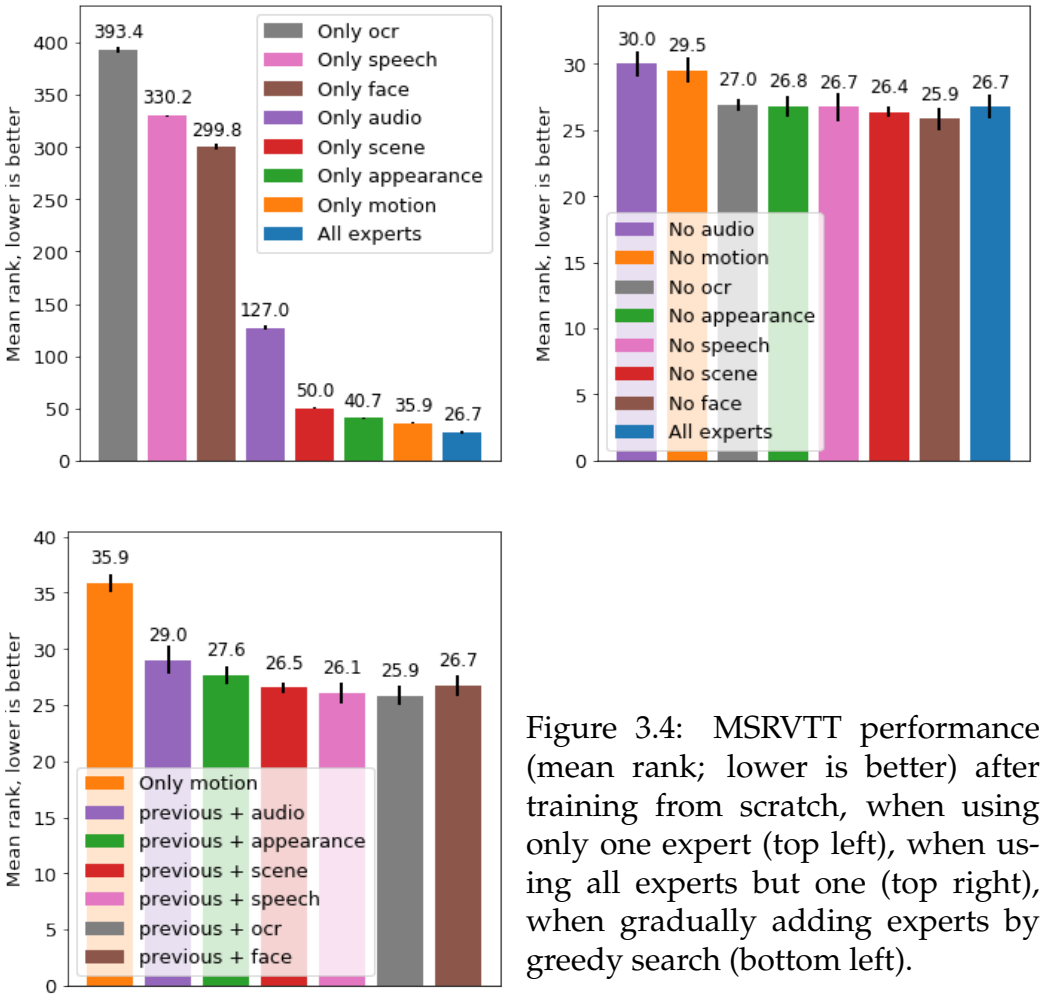

Figure 3.4: MSRVTT performance (mean rank; lower is better) after training from scratch, when using only one expert (top left), when using all experts but one (top right), when gradually adding experts by greedy search (bottom left).

state-of-the-art results on all the three datasets. The gains obtained through MMT's long term temporal encoding are particularly noticeable on the long videos of ActivityNet.

3.3.4 Model complexity

Number of parameters. As shown below, using multiple modalities does not impact the number of parameters significantly. Interestingly, majority of the parameters correspond to the BERT caption encoding module. We also note that the difference in the video encoder comes from the projections. The number of parameters of a transformer encoder is independent

Table 3.4: Retrieval performance on the MSRVTT dataset. 1k-A and 1k-B denote test sets of 1000 randomly sampled caption-video pairs used in [162] and [103] resp.

		Text ⟶ *Video*			*Video* ⟶ *Text*		
Method	Split	R@5↑	MdR↓	MnR↓	R@5↑	MdR↓	MnR↓
Random baseline	1k-A	0.5	500.0	500.0	0.5	500.0	500.0
JSFusion [162]	1k-A	31.2	13	-	-	-	-
HT [104]	1k-A	35.0	12	-	-	-	-
CE [90]	1k-A	$48.8_{\pm 0.6}$	$6.0_{\pm 0.0}$	$28.2_{\pm 0.8}$	$50.3_{\pm 0.5}$	$5.3_{\pm 0.6}$	$25.1_{\pm 0.8}$
Ours	1k-A	$\mathbf{54.0}_{\pm 0.2}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.9}$	$\mathbf{56.0}_{\pm 0.9}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{23.6}_{\pm 1.0}$
HT-pretrained [104]	1k-A	40.2	9	-	-	-	-
Ours-pretrained	1k-A	$\mathbf{57.1}_{\pm 1.0}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{24.0}_{\pm 0.8}$	$\mathbf{57.5}_{\pm 0.6}$	$\mathbf{3.7}_{\pm 0.5}$	$\mathbf{21.3}_{\pm 0.6}$
Random baseline	1k-B	0.5	500.0	500.0	0.5	500.0	500.0
MEE [103]	1k-B	37.9	10.0	-	-	-	-
JPose [154]	1k-B	38.1	9	-	41.3	8.7	-
MEE-COCO [103]	1k-B	39.2	9.0	-	-	-	-
CE [90]	1k-B	$46.0_{\pm 0.4}$	$7.0_{\pm 0.0}$	$35.3_{\pm 1.1}$	$46.0_{\pm 0.5}$	$6.5_{\pm 0.5}$	$30.6_{\pm 1.2}$
Ours	1k-B	$\mathbf{49.1}_{\pm 0.4}$	$\mathbf{6.0}_{\pm 0.0}$	$\mathbf{29.5}_{\pm 1.6}$	$\mathbf{49.4}_{\pm 0.4}$	$\mathbf{6.0}_{\pm 0.0}$	$\mathbf{24.5}_{\pm 1.8}$

of the number of input embeddings, as are the parameters of a CNN from the image size.

Our cross-modal architecture using 7 modalities has: 133.3M parameters, including caption encoder: 112.9M, video encoder: 20.4M (Projections: 3.3M, MMT: 17.1M). Our cross-modal architecture using 2 modalities has: 127.3M parameters, including caption encoder: 109.6M (decrease compared to 7 modalities due to using less gated embedding modules), video encoder: 17.7M (Projections: 0.6M, MMT: 17.1M).

Training and inference times. Training our full cross-modal architecture from scratch on MSRVTT takes about 4 hours on a single V100 16GB GPU.

If we replace our multi-modal transformer by collaborative gating [90], we reduce the number of parameters from 133.3M to 123.9M. However, the gain in inference time is minimal, from 1.1s to 0.8s, and is negligible compared to feature extraction, as detailed below.

Inference time for 1k videos and 1k text queries from MSRVTT on a single V100 GPU is as follows: approximately 3000s to extract features of 7 experts on 1k videos (480s just for S3D motion features), 1.1s to process

Table 3.5: Retrieval performance on the ActivityNet dataset.

	Text $\longrightarrow$ *Video*			*Video* $\longrightarrow$ *Text*		
Method	R@5↑	MdR↓	MnR↓	R@5↑	MdR↓	MnR↓
Random baseline	0.1	2458.5	2458.5	0.1	2458.5	2458.5
FSE [166]	$44.8_{\pm 0.4}$	7	-	$43.1_{\pm 1.1}$	7	-
CE [90]	$47.7_{\pm 0.6}$	$6.0_{\pm 0.0}$	$23.1_{\pm 0.5}$	$46.6_{\pm 0.7}$	$6.0_{\pm 0.0}$	$24.4_{\pm 0.5}$
HSE [166]	49.3	-	-	48.1	-	-
Ours	$\mathbf{54.2}_{\pm 1.0}$	$\mathbf{5.0}_{\pm 0.0}$	$\mathbf{20.8}_{\pm 0.4}$	$\mathbf{54.8}_{\pm 0.4}$	$\mathbf{4.3}_{\pm 0.5}$	$\mathbf{21.2}_{\pm 0.5}$
Ours-pretrained	$\mathbf{61.4}_{\pm 0.2}$	$\mathbf{3.3}_{\pm 0.5}$	$\mathbf{16.0}_{\pm 0.4}$	$\mathbf{61.1}_{\pm 0.2}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{17.1}_{\pm 0.5}$

Table 3.6: Retrieval performance on the LSMDC dataset.

	Text $\longrightarrow$ *Video*			*Video* $\longrightarrow$ *Text*		
Method	R@5↑	MdR↓	MnR↓	R@5↑	MdR↓	MnR↓
Random baseline	0.5	500.0	500.0	0.5	500.0	500.0
CT-SAN [163]	16.3	46	-	-	-	-
JSFusion [162]	21.2	36	-	-	-	-
CCA [78] (rep. by [103])	21.7	33	-	-	-	-
MEE [103]	25.1	27	-	-	-	-
MEE-COCO [103]	25.6	27	-	-	-	-
CE [90]	$26.9_{\pm 1.1}$	$25.3_{\pm 3.1}$	-	-	-	-
Ours	$\mathbf{29.2}_{\pm 0.8}$	$\mathbf{21.0}_{\pm 1.4}$	$\mathbf{76.3}_{\pm 1.9}$	$\mathbf{29.3}_{\pm 1.1}$	$\mathbf{22.5}_{\pm 0.4}$	$\mathbf{77.1}_{\pm 2.6}$
Ours-pretrained	$\mathbf{29.9}_{\pm 0.7}$	$\mathbf{19.3}_{\pm 0.2}$	$\mathbf{75.0}_{\pm 1.2}$	$\mathbf{28.6}_{\pm 0.3}$	$\mathbf{20.0}_{\pm 0.0}$	$\mathbf{76.0}_{\pm 0.8}$

videos with MMT, 0.9s to process 1k captions with BERT+gated embedding modules, 0.05s to compute similarities and rank the video candidates for the 1k queries.

3.4 Conclusion

We introduced multi-modal transformer, a transformer-based architecture capable of attending multiple features extracted at different moments, and from different modalities in video. This leverages both temporal and cross-modal cues, which are crucial for accurate video representation. We incorporate this video encoder along with a caption encoder in a cross-modal framework to perform caption-video matching and obtain state-of-the-art results for video retrieval.

Chapter 4

Pretraining a model on all video modalities

We live in a multi-modal world, communicating through speech, visual signals and sound. This is reflected in the videos created and uploaded online—often they are accompanied by a highly informative audio track containing cues complementary to visual content. Our goal in this chapter is to retrieve audio-visual videos with natural language queries.

While many popular video understanding works [17,80,86,102,123,159] restrain the video signal to a sequence of visual frames, several approaches [50, 90, 103] have progressed to incorporate information from different modalities through the use of pretrained feature extractors called “experts". For videos, naturally composed of multimodal information, learning the optimal fusion of different modality ‘experts’ is paramount. This challenge of multimodal learning is made more difficult by the scarcity of large manually-captioned video datasets. Existing datasets, e.g., [80, 87, 130, 159, 171] remain small scale. This has led to several approaches utilising the large amount of instructional videos online [50, 102, 104, 123], where transcribed speech (obtained with ASR) is closely linked to visual content, and hence a valuable source of supervision to train video encoders. Because of the proximity between text queries and speech, this approach presents the advantage of transferring well to text-to-video retrieval tasks. However, because the speech modality is used as a source of ‘pseudo’ captioning labels, most of these works [50, 102, 104] only pretrain an encoder to process non-speech modalities (RGB, audio, etc), thereby not learning to combine speech and visual inputs effectively during pretraining. For many videos online, effectively processing speech is crucial for accurate video retrieval (Fig. 4.1).

In this chapter, we propose a novel pretraining strategy for learning multi-modal fusion from instructional videos (Fig. 4.2, top right). We learn two encoders - the first being a video encoder (Ψ) that fuses experts from

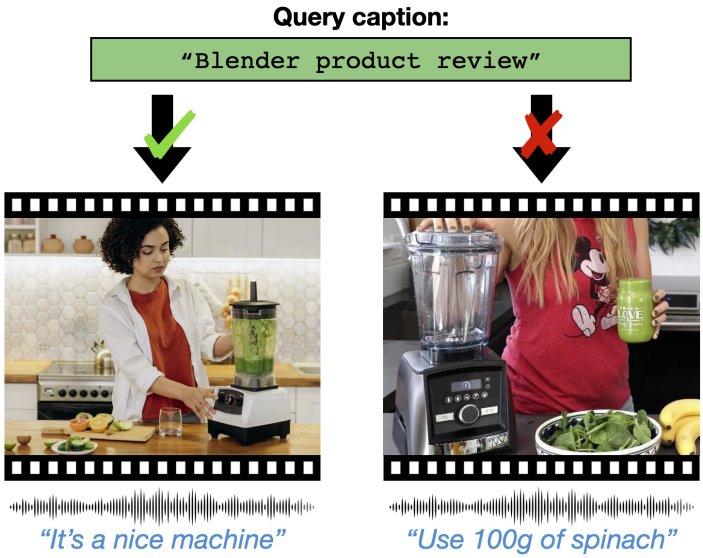

Figure 4.1: Speech is part of the story! Video retrieval methods that focus on visual inputs alone are likely to miss out on key information (e.g., while both the examples above contain a blender, the speech (in blue) helps identify the one for a product review). In this work, we focus on learning a video encoder to effectively process RGB and audio features, as well as transcribed speech from instructional videos online, through a novel modality masking method. Our approach learns from unlabelled videos, without the need for expensive manual captions.

three modalities - RGB, transcribed speech (which we henceforth refer to as ASR for brevity), and audio. During pretraining we use a modality masking strategy, where we mask out an entire modality in the input of the video encoder, and try to predict an encoded version of this modality (encoded using a second encoder (Φ)) from the other modalities. In this manner, the modality being predicted is effectively being used as 'supervision' for the other two. At each batch, we mask a different modality, thereby learning a video encoder that is able to process all the modalities available in the video signal.

We make the following two contributions: (i) We introduce a new pretraining approach for learning video representations that does not require costly manual annotations. Unlike previous works [50, 87, 102, 104], we train our video encoder with three inputs – RGB, audio and ASR, and alternate at each batch which one is used for supervision. At the time of

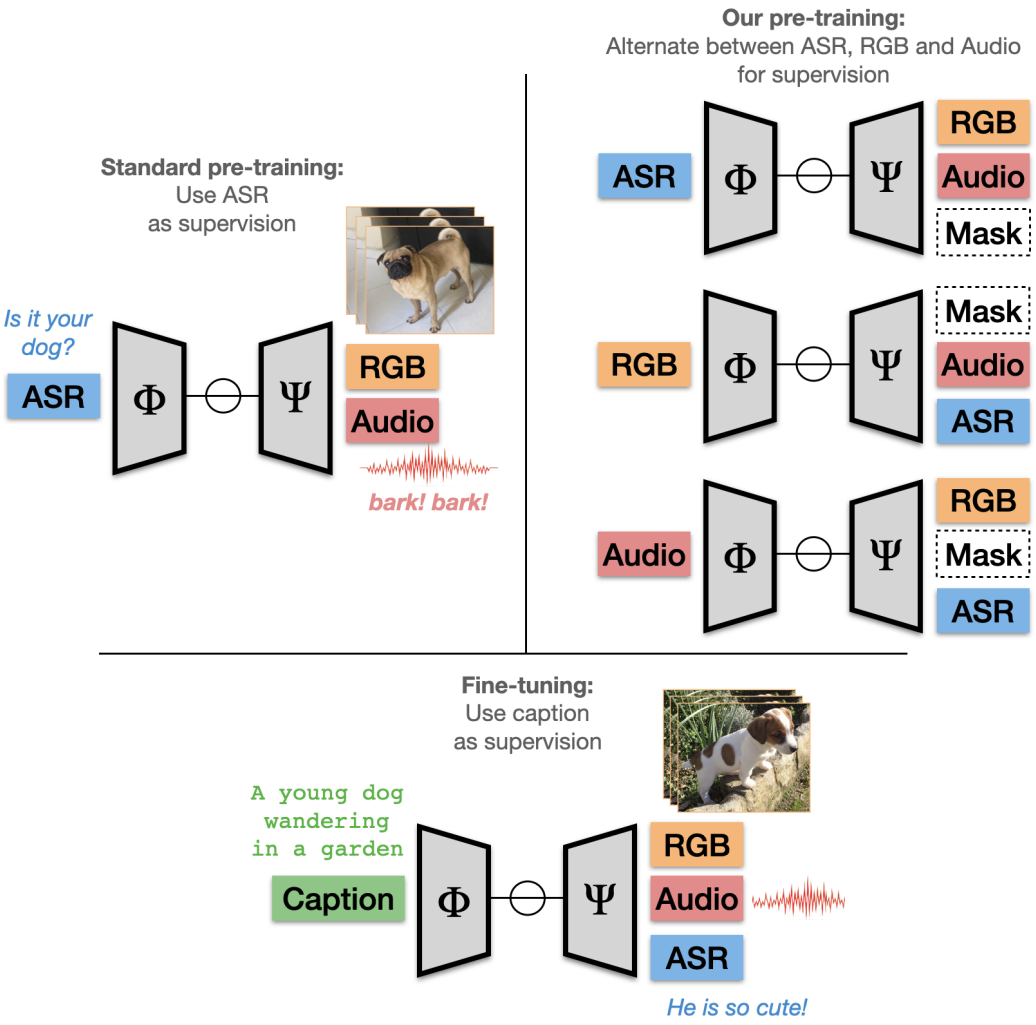

Figure 4.2: A common paradigm in learning from instructional videos is use transcribed speech (from ASR) (projected here using an encoder Φ) to supervise a video encoder Ψ (top left). Instead, we train our video encoder Ψ with three inputs – RGB, audio and transcribed speech (ASR), and alternate between masking and predicting an entire modality at a time (top right). At the time of finetuning (bottom), our video encoder has been pretrained to use all video modalities.

finetuning, our video encoder has been pretrained to use all video modalities. (ii) We obtain competitive results on several standard text-to-video benchmarks.

4.1 Related work

4.1.1 Video datasets for audio-visual retrieval

Despite the fact that videos are often inherently multimodal, many popular works [17, 80, 86, 102, 123, 159] for video and language understanding discard the audio signal, potentially losing rich and varied additional information mentioned in speech or other background sounds. This may be due to the difficulty of jointly processing multiple modalities and the computational cost associated with processing such a high dimensional signal. Another factor may also be an inherent bias towards the visual modality in the annotation procedure for many datasets. For example, the LSMDC dataset [130] is annotated with audio descriptions (AD), which by nature must ignore the speech track. In the ActivityNet dataset [80], annotators were explicitly asked to ignore the audio track. The MSVD dataset [28] videos simply do not have audio. In this work, we pretrain a video encoder using multiple video modalities, including audio and speech. We therefore also evaluate our approach on datasets such as How2R [87], CMD [16] and YouCook2 [171], where speech plays an important role in understanding video content.

4.1.2 Pretraining for Video and Language

Since the release of the HowTo100M dataset [104], a large instructional video dataset, there has been a renewed interest in leveraging large-scale pretraining to improve video-text representations for tasks such as video question-answering [87, 141], text-to-video retrieval [50, 104, 123, 158], action recognition [7, 102, 111, 147] and video captioning [70, 172].

In NLP, BERT [40] and its variants have popularized the 'masked language modelling' self-supervised technique for pretraining, wherein words in the input are randomly masked and the training objective is tasked with predicting their encodings. This technique has been extended to train visual and language encoders (eg. VideoBERT [148], CBT [147], ViLBERT [93], Hero [87] etc). All such works, only mask a proportion of the input (usually 15%). Contrary to natural language, the visual signal and audio signal of a video are continuous and highly redundant. A masked video or audio segment can be easily estimated from its neighboring frames. To address this problem, we mask out an entire modality in the input, forcing our model to learn difficult cross-modal interactions. We do not estimate the masked signal directly but use a contrastive objective across the training batch.

Patrick et al. [123] have proposed to mitigate some issues of contrastive objectives using a cross-captioning loss. In [122], the authors generalize the contrastive self-supervised learning approach to multi-modal data. They show that the process of forming positive and negative pairs correspond to a composition of transformations that should be chosen and combined methodically. To this end, they extend the image self-supervised learning approach to the video domain by introducing temporal transformation like time-reversal and time-shifting. Contrary to them, we tackle multi-modal representation of videos across vision, audio and speech.

4.2 Methodology

In this section, we first describe the common pretraining approach for learning from instructional videos, where the ASR is used to supervise a visual encoder. We then present our strategy to pretrain a video encoder Ψ on three video modalities: RGB, Audio and ASR, by using each of them to supervise the others in an alternating manner. After pretraining, our video encoder has learnt to attend across all modalities in a video, and can be finetuned on video-text datasets for the task of video retrieval.

4.2.1 Standard Pretraining

As a video representation learning pretraining strategy, several previous works [50, 102, 104] use the speech modality as supervision to train a video encoder on the other video modalities. Illustrated on the top left side of Fig. 4.2, this approach involves the estimation of a speech representation by a query encoder Φ and a video representation by a video encoder Ψ. The training objective is usually a standard metric learning objective (maximising the similarity between the speech representation and the video representation if they are extracted from the same video, minimizing the similarity between randomly selected speech and video). At the time of finetuning (Fig. 4.2, bottom, the query encoder Φ is used to encode the caption while the video encoder Ψ is processing all video modalities, including speech.

The main drawback of this approach is that the video encoder is not pretrained on speech since that modality is used as pretraining supervision. At the end of pretraining, the video encoder has hence been denied the opportunity to learn complex cross-modal interactions between RGB

and speech. The video encoder only learns to process speech during finetuning. This is a major limitation as speech may be an integral part of the video signal and encode crucial information for video retrieval.

4.2.2 Alternating Modality-Masking Pretraining

We propose a new approach for pretraining a video encoder on a large-scale dataset of raw videos like HowTo100M [104], which does not contain captioning labels. In order for our video encoder to be pretrained on all video modalities, **including ASR**, we propose to not only use ASR supervision, but to alternate between three objectives (Fig. 4.2, top right):

1. Use ASR as supervision to train the video encoder Ψ on processing RGB + Audio as inputs
2. Use RGB as supervision to train the video encoder Ψ on processing Audio + ASR as inputs
3. Use Audio as supervision to train the video encoder Ψ on processing RGB + ASR as inputs

At each training batch, we randomly pick one of those objectives. We therefore randomly pick a modality in $\{RGB, Audio, ASR\}$ to serve as the supervising modality processed by the query encoder, while the other two modalities act as the collaborating modalities processed by the video encoder. Let us take the example of a training batch for which RGB has been selected as the supervising modality. For each video of the batch, its sequence of RGB features will be processed by the query encoder Φ to obtain a query representation. The features of the other two modalities of the video, in this case Audio and ASR, will be processed by the video encoder Ψ to extract both cross-modal and temporal information and obtain a video representation. We will then proceed to optimize the parameters of our encoders so that the query and video representations of a same video are similar while the representations of different videos in the batch are dissimilar.

More formally, for each video clip v_i in the training batch, we separate the expert features in two sets: q_i are the features obtained from the supervising modality and c_i are the features obtained from the collaborating modalities. We then use our query encoder Φ to compute a representation $\Phi(q_i)$ of the supervising modality. Similarly, our video encoder Ψ will compute a representation $\Psi(c_i)$ of the collaborating modalities.

During finetuning (Fig.4.2, bottom), our video encoder Ψ is provided with all the modalities present in the video signal, all of which it has seen before during pretraining, and has hence acquired the ability to model cross-modal complex correlations.

Although our video encoder Ψ only ever receives two modalities at a time during pretraining, they are different at each batch. We therefore need a video encoder capable of processing the three video modalities, but at each batch, one of them (the supervising modality) is "masked out", it is simply not provided to Ψ. In the next section we describe the architecture of that encoder.

4.2.3 The Multi-Modal Transformer

For our video encoder Ψ, we use the Multi-Modal Transformer described in [50]. It consists in a Transformer encoder that is fed features from different video modalities. The self-attention mechanism of the Transformer allows each token to attend to all the others, therefore being able to process information across both time and modalities. The choice of the MMT architecture for our modality-masking pretraining approach is justified by its capacity to elegantly handle missing modalities. In fact, all the transformer layers parameters are shared across all input features, and therefore modalities. That means that even if one modality is masked from the input of MMT, the parameters of all layers will still be optimized. All parameters needed for the downstream task are optimized at each batch, independently of the chosen objective. This is in contrast to the MoEE style of architecture [103] where there is a dedicated encoding branch for each modality. In the case of a missing expert stream, zeros will be fed, thereby wasting computation for that whole branch.

4.2.4 Loss Function

For both pretraining and finetuning, we optimize both query encoder Φ and video encoder Ψ to provide similar representations when their input features come from the same video clip and dissimilar representations when they come from different clips. We train our model with the bi-directional max-margin ranking loss [75]:

$$\mathcal{L} = \frac{1}{B} \sum_{i=1}^{B} \sum_{j \neq i} \Big[\max(0, s_{ij} - s_{ii} + m) + \max(0, s_{ji} - s_{ii} + m) \Big], \tag{4.1}$$

where B is the batch size, $s_{ij} = s(\Phi(q_i), \Psi(c_j))$, the similarity score between query representation $\Phi(q_i)$ of video v_i and video representation $\Psi(c_j)$ of video v_j, and m is the margin. This loss enforces the similarity between true representation pairs s_{ii} to be higher than the similarity between negative samples s_{ij} or s_{ji}, for all $i \neq j$, by at least m. This will have the effect of gathering similar captions and videos together in the embedding space, thereby allowing video retrieval to be performed by ranking videos according to their proximity with the query.

4.2.5 Selection of Modalities

We choose the modalities RGB, Audio and ASR in this work, largely because they often represent complementary aspects of the video signal. Although audio and speech are both extracted from the audio signal, the expert models extracting features for those modalities have been pretrained on different tasks and present specialized architectures dedicated to those tasks. The CNN used to encode the audio (sounds) features has not been trained on ASR, and therefore does not embed speech semantics. Similarly, the speech encoder is only provided with the transcript words, hence not encoding information about the background audio sounds.

4.3 Experiments

We first describe the text-video datasets that our model is trained and evaluated on (sec. 4.3.1), then present implementation details (sec. 4.3.2) and ablation studies (sec. 4.3.3). Finally, we compare to the state-of-the-art for video retrieval (sec. 4.3.4).

4.3.1 Datasets and Metrics

Since the focus of our work is the effective encoding of ASR, audio and visual information, we evaluate on video datasets that contain multimodal captions (i.e., captions that refer to the content in the speech as well). For each dataset, we manually inspect 100 caption-video pairs at random to determine the percentage of captions that are related to what is being said in the video. For example, the caption “Someone talking about love" requires knowledge of the speech in the video, whereas “A woman with a red dress" does not. Results are reported below.

HowTo100M [104] is a very large-scale dataset of over 1M YouTube instructional videos that amounts to about 15 years of video. This dataset

was not manually annotated with captions, but is a valuable source of data for self-supervised learning because of the high correlation between visual, audio and speech information in its videos. We only use this dataset to pretrain our model.

How2R [87] features 47,369 clips extracted from the HowTo100M dataset videos and split into a training, validating and testing set. The clips are 17s long on average, and annotated with a caption. Our manual inspection of 100 captions yields 54 captions related to speech. Because it has the same domain as our pretraining dataset, we run ablation studies on this dataset.

MSR-VTT [159] contains 10K YouTube videos with 200K descriptions. Following other works [90], we train on 9K train+val videos and report results on the 1K-A test set. After manual inspection, we find that 12% of the captions are related to speech.

Condensed Movies Dataset (CMD) [16] consists of 33,976 clips extracted from 3,605 movies. Our manual verification process indicates that approximately 60% of CMD descriptions are related to speech.

YouCook2 [171] consists of 176 hours of cooking videos. The videos are segmented into 13,829 clips, each annotated with a sentence describing a step of the recipe. We follow [104] and evaluate our model on 3,350 clips that are not present in HowTo100M. We found about 70% of YouCook2 captions are related to the speech in the video. For example, in the case of a video annotated with "add corn starch", paying attention to the speech: "I now use corn starch" is a strong cue indicating that we are not dealing with flour.

ActivityNet captions [80] consists of 20K YouTube videos annotated with several sentences. We follow [166] and concatenate the sentences to obtain a paragraph annotation for each video. We found that only 1% of the captions relate to speech. The authors of this dataset informed us that the annotators were explicitly asked to ignore the audio. It was turned off by default for the annotation process.

Metrics. We evaluate the performance of our approach on the following standard retrieval metrics: recall at rank N (R@N, higher is better), median rank (MdR, lower is better) and mean rank (MnR, lower is better). For each metric we run the experiment with 3 random seeds and report the mean and standard deviation. We report the test-set performances of our model for the epoch where the validation-set geometric mean of R@1, R@5 and R@10 is maximal.

4.3.2 Implementation Details

Pretrained experts. Both encoders use pretrained expert models for extracting features from each video modality. We use the following 3 experts:
RGB features are extracted from S3D [156] trained on the Kinetics action recognition dataset. We extract one RGB feature of dimension 1024 per second of video.
Audio features are extracted using VGGish model [66] trained on the YouTube8M dataset [1]. We extract one audio feature of dimension 128 per second of video.
ASR transcripts are obtained from the closed captions accompanying videos on YouTube. Words are encoded with BERT-base-cased [40]. We obtain one speech feature of dimension 768 for each wordpiece from the ASR.

Query encoder Φ for pretraining. In the case when the masked modality is either RGB or audio, we encode it using the Multi-Modal-Transformer (MMT) [50] model. We follow [50] and use a 4 layer, 4 head version of MMT. For any modality presented to the query encoder, we do not encode input sequence of features with temporal embeddings. We found that this made the pretraining objective trivially easy to solve - we hypothesise that this is because temporal information allows the encoders to align silences in the ASR and audio features (as both are extracted from the same audiotrack). For example, both encoders would be able to determine the presence and absence of speech from the ASR and audio modalities. The similarity can then be maximised based on this temporal alignment, instead of on the video semantics, leading to performance drops. In the case when the masked modality is ASR, we follow [50] and process the speech words with a pretrained BERT model. For memory constraints, we limit BERT input to 30 consecutive wordpieces, randomly sampled from the ASR. The representation extracted from the BERT [CLS] token is projected by 3 different gated embedding units (one for each modality) to obtain our query representation $\Phi(q_i)$.

Query encoder Φ for finetuning. During finetuning, we use the captions as supervision. For finetuning on MSRVTT, YouCook2 and ActivityNet, we follow the procedure introduced in MMT [50]: we process the caption with the Bert-based query encoder that we pretrained earlier. We limit BERT input to 30 consecutive wordpieces, randomly sampled in the caption. On the How2R and CMD datasets, we found out that using the

pretrained Bert-based query encoder for encoding the captions resulted in rapid over-fitting – on the other hand, freezing the weights in the query encoder lead to poor performance. We therefore follow the approach outlined in MoEE [103] and use a net-VLAD layer [9] to aggregate the caption word embeddings (obtained by a frozen BERT model), to obtain the final caption representation. The caption representation is then projected by 3 different gated embedding units.

Video encoder Ψ. This is implemented using the Multi-Modal Transformer (MMT) [50] as our video encoder. We use a 4 layers x 4 heads version of MMT with a dropout probability of 10%, a hidden size d_{model} of 512, and an intermediate size of 3072. We initialize the aggregated embeddings of MMT with a max-pooling aggregation of the modality features. In the case of a masked modality (pretraining) or when a modality is not available in the video, no features are provided to MMT and the aggregated embedding for that modality becomes a zero vector. For all our experiments, we only use the sequences of features extracted by our RGB, audio and ASR pretrained experts. The parameters of those feature extractors are kept frozen. For memory constraints, we provide the video encoder with sequences of maximum 30 features for the RGB and audio modalities, and maximum 128 features for the ASR. In case more features are available in the video, they are randomly sampled.

Hyperparameters. For each dataset, we estimate the hyperparameters by running a grid search on the corresponding validation set. We use the Adam optimizer for all the experiments.

For pretraining on HowTo100M, we use a batch size of 1,200 videos, an initial learning rate of 1e-4, which we decay by a 0.98 multiplicative factor every 2K optimisation steps, and train for 400K steps. We randomly crop HowTo100M videos into segments of 30 seconds.

For training from scratch or finetuning on MSR-VTT or YouCook2, we use a batch size of 32 videos, an initial learning rate of 5e-5, which we decay by a 0.95 multiplicative factor every 1K optimisation steps, and train for 50K steps. For training from scratch or finetuning on CMD or How2R, we use an initial learning rate of 5e-5, which we decay by a 0.90 multiplicative factor every 375 optimisation steps, and train for 20K steps. We use a batch size of 32 videos on How2R and 64 videos on CMD. For training from scratch or finetuning on ActivityNet, we use a batch size of 24 videos, an initial learning rate of 5e-5, which we decay by a 0.90 multiplicative factor every 1K optimisation steps, and train for 50K steps.

For training on HowTo100M, MSR-VTT, YouCook2 or ActivityNet, the bidirectional max-margin ranking loss margin is set to 0.05. For training on How2R or CMD, it is set to 0.2.

Running time. Pretraining our model on HowTo100M takes 12 days on 8 V-100 GPUs. Finetuning on MSRVTT, How2R, CMD, YouCook2 or ActivityNet takes about 4 hours on a single V-100 GPU.

4.3.3 Ablation Analysis

We perform three ablation studies to: (i) show the effect of varying the masking probability of ASR, p, during pretraining; (ii) demonstrate the need of complete modality masking over partial modality masking; and (iii) compare multi-modal retrieval results to those with a single modality.

Effect of the ASR masking probability p. Table 4.1 shows the impact of different masking probabilities during pretraining on HowTo100M. The probability p refers to the probability of masking our ASR and feeding in only RGB and audio to the candidate encoder (this is the common pretraining paradigm, where ASR is effectively 'supervising' our video encoder). The rest of the time is equally split between masking out audio and RGB. Hence if $p = 0.8$, we mask out ASR 80% of the time, RGB 10% of the time, and audio 10% of the time. Note that this is equivalent to weighting the loss (Eq. 4.1) differently depending on which modality is masked. We report results on the validation set of How2R after finetuning on the training set of How2R. We show that the common pretraining paradigm of always using the ASR to supervise RGB and audio ($p = 1.0$, first line) does not provide the best results. It is better to also use audio and RGB as supervision in order to pretrain the video encoder on speech. For the rest of the experiments, we set $p = 0.8$ during pretraining.

Advantage of complete modality masking over partial masking. Several recent works [87,93,147,148] pretrain a video encoder by partially masking a modality (eg: masking 15% of video frames). We instead mask 100% of the modality. We have compared our approach with masking 15%, 50% or 85% of the supervising modality tokens. To not make the task trivial we did not provide the query encoder Φ with 100% of the supervising modality tokens, but only with the feature tokens that were masked from the video encoder Ψ. We used the setting which obtained best results for 100%

Table 4.1: The effect of the masking probability for transcribed speech (ASR) p, where $p = 1.00$ refers to the case where ASR is masked 100% of the time, and predicted from audio and RGB. Results are reported on the validation set of How2R after finetuning. We note that performance improves when $p < 1$, but remains relatively robust to different values.

	Text $\Longrightarrow$ Video				
p	R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
1.00	$3.1_{\pm 0.1}$	$9.9_{\pm 0.3}$	$15.5_{\pm 0.3}$	$97.0_{\pm 2.2}$	$292.3_{\pm 4.7}$
0.90	$2.9_{\pm 0.0}$	$9.5_{\pm 0.0}$	$15.2_{\pm 0.0}$	$96.5_{\pm 0.0}$	$271.9_{\pm 0.0}$
0.80	$\mathbf{3.7}_{\pm 0.1}$	$\mathbf{11.5}_{\pm 0.1}$	$\mathbf{17.8}_{\pm 0.3}$	$\mathbf{79.0}_{\pm 0.0}$	$270.7_{\pm 2.5}$
0.70	$3.5_{\pm 0.1}$	$\mathbf{11.5}_{\pm 0.2}$	$17.6_{\pm 0.1}$	$80.7_{\pm 0.9}$	$\mathbf{267.8}_{\pm 1.4}$
0.33	$3.5_{\pm 0.2}$	$\mathbf{11.5}_{\pm 0.3}$	$\mathbf{17.8}_{\pm 0.2}$	$82.0_{\pm 1.4}$	$269.8_{\pm 1.5}$

masking, i.e., ASR is used as supervision for 80% of the batches, audio 10% and RGB 10%.

Recall@10 results on the validation set of How2R are: From scratch (no pretraining): 12.9, Masking 15%: 16.1, Masking 50%: 16.2, Masking 85%: 16.8, Masking 100%: 17.8.

The results for the partial masking pretraining show a lower performance compared to 100% masking. We also noticed that during pretraining the loss for the partial masking experiments was lower than the loss for the 100% masking experiment. This can be attributed to the fact that the query encoder Φ and video encoder Ψ are both provided with some of the supervising modality features, making the pretraining task easier, and therefore less effective. This is particularly the case for the audio and visual modality because of their continuity and high redundancy.

Impact of pretraining on single-modality retrieval. We further evaluate our pretraining approach by finetuning the model on a single video modality. In this case, for each video in the How2R validation set, our video encoder is only provided with the features of one modality, either RGB, Audio or ASR. We report the results using R@10 in Fig. 4.3. When only pretraining with ASR supervision ($p = 1.0$, orange), our video encoder only processes RGB and audio inputs. In this case, we note that pretraining helps when finetuning only on the RGB modality, but not on the audio modality. As expected, this setting leads to a performance drop on ASR, as the video encoder has never seen ASR inputs during pretraining. This

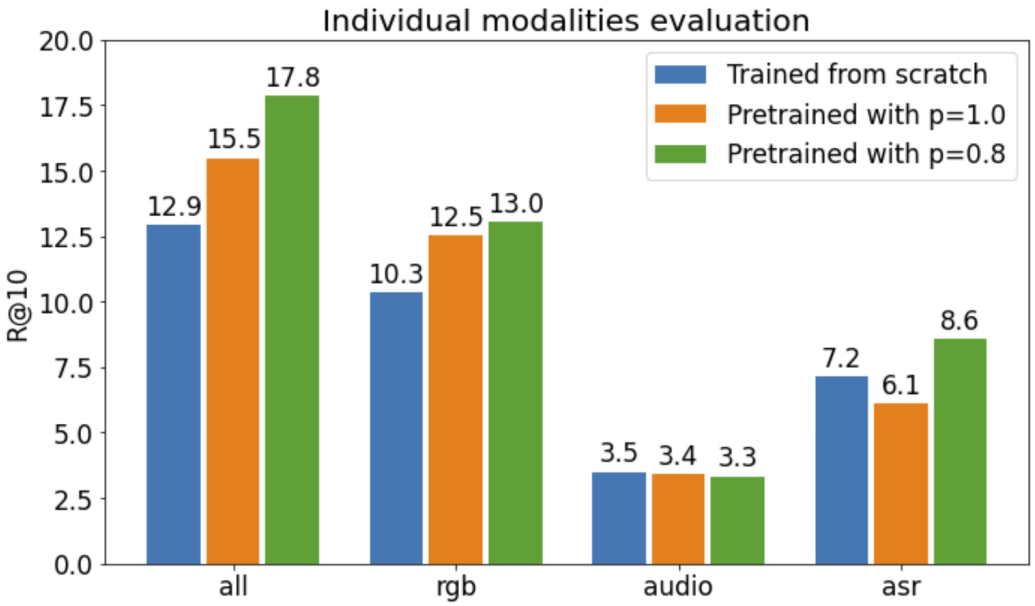

Figure 4.3: Impact of the pretraining approach on the retrieval of a single modality. We report results on the val set of How2R using R@10. (Best viewed in colour.)

is not the case for our alternating modality masking approach where the video encoder was sometimes provided with ASR features and therefore learns to process that modality. We note that our alternating masking approach ($p = 0.8$, green) provides improvements overall, as well as for each modality independently (other than for audio which does not seem to benefit from pretraining).

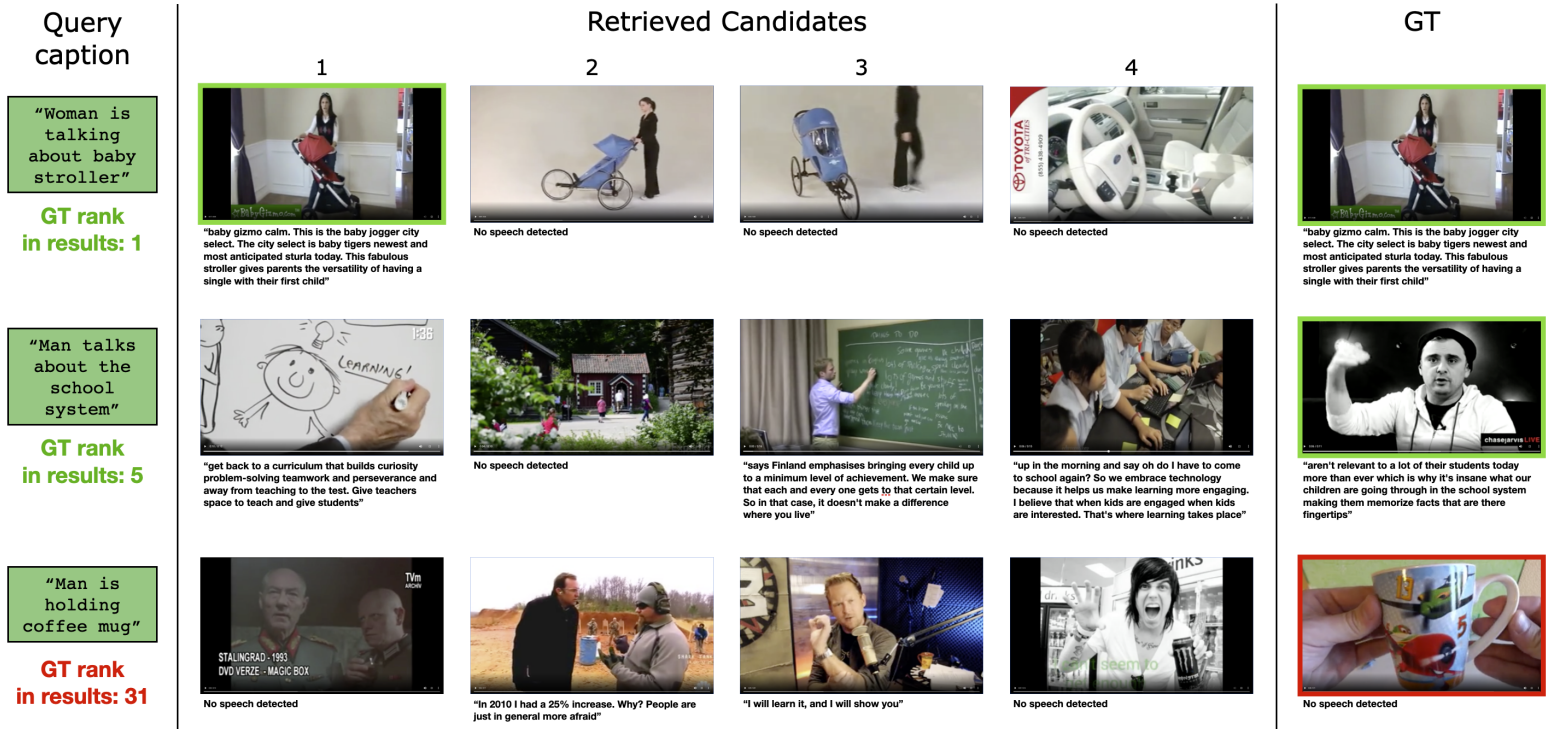

Figure 4.4: Qualitative results of our retrieval method on the MSR-VTT dataset. For each query, we show frames and ASR from the top 4 ranked videos as well as for the ground truth video. We indicate the rank of the ground-truth video in our retrieval results (highlighted in green when it is in the top-5 retrieved results, or red otherwise) on the left under the query. Note that there are 1000 candidate videos in the test set. (Best viewed on screen.)

4.3.4 Comparison to the State of the Art

Results on How2R are provided in Table 4.2. The original paper introducing the How2R dataset [87] tackles the task of moment localization in a video clip. We re-purpose the How2R dataset for the task of video retrieval where each moment and its description are considered as a different video-caption pair. We reproduce the MoEE approach [103] on this dataset, and show that our method trained from scratch significantly outperforms MoEE. We also implement the MMT pretraining approach [50] (equivalent to p=1.0) with our features, and compare it with our modality masking pretraining (p=0.8) approach. The large performance improvement obtained with our approach demonstrates the advantage of pretraining the video encoder on speech before finetuning on the How2R dataset, which has more than half of its captions related to speech.

Table 4.2: Text to Video retrieval results on the How2R [87] benchmark. † Our implementation on this dataset using only our RGB, audio and ASR features. sc: trained from scratch on How2R. pt: pretrained on the HowTo100M dataset, then finetuned on How2R.

	Text $\Longrightarrow$ Video				
Method	R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
Random	0.0	0.1	0.2	2009.5	2009.5
MoEE (sc) [103]†	$2.2_{\pm 0.1}$	$7.8_{\pm 0.1}$	$12.9_{\pm 0.3}$	$118.7_{\pm 1.2}$	$389.5_{\pm 1.8}$
Ours (sc)	$2.3_{\pm 0.2}$	$8.3_{\pm 0.3}$	$13.6_{\pm 0.2}$	$106.0_{\pm 2.2}$	$312.5_{\pm 1.2}$
MMT (pt) [50]†	$2.9_{\pm 0.0}$	$9.1_{\pm 0.2}$	$14.5_{\pm 0.2}$	$96.0_{\pm 2.2}$	$314.3_{\pm 1.7}$
Ours (pt p=0.8)	$\mathbf{3.4}_{\pm 0.2}$	$\mathbf{11.6}_{\pm 0.2}$	$\mathbf{18.2}_{\pm 0.3}$	$\mathbf{75.3}_{\pm 0.9}$	$\mathbf{277.1}_{\pm 2.3}$

Results on CMD are provided in Table 4.3. Unlike the original CMD paper [16], we remove actor names from the captions. We re-implement MoEE [103] on this modified dataset using our features, and demonstrate that our pretraining approach provides a significant improvement in performance. Note that this is despite the large variation in domain between pretraining and finetuning – while we pretrain on instructional videos from YouTube, CMD consists of short clips extracted from movies.

Table 4.4 presents results on YouCook2. Due to the high importance of the speech modality in this dataset, pretraining with our approach (pt p=0.8) yields considerable performance improvement, compared to the standard pretraining approach (MMT) that does not pretrain the video encoder on the speech modality.

Table 4.3: Results on the Condensed Movies Dataset (CMD) [16]. † Our implementation on this dataset using only our RGB, audio and ASR features. ‡ Our implementation on this dataset using the code and all the features provided by the authors of CMD [17]. sc: trained from scratch on CMD. pt: pretrained on the HowTo100M dataset, then finetuned on CMD.

	Text ⟹ Video				
Method	R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
Random	0.0	0.1	0.2	3284.5	3284.5
MoEE (sc) [103]†	$3.2_{\pm 0.1}$	$9.9_{\pm 0.3}$	$14.9_{\pm 0.4}$	$142.7_{\pm 0.5}$	$532.7_{\pm 5.7}$
CMD (sc) [16]‡	2.6	10.2	16.2	102	377.7
Ours (sc)	$4.6_{\pm 0.1}$	$13.5_{\pm 0.2}$	$19.5_{\pm 0.1}$	$89.7_{\pm 1.2}$	$396.5_{\pm 5.5}$
Ours (pt p=0.8)	$\mathbf{5.8}_{\pm 0.2}$	$\mathbf{15.8}_{\pm 0.2}$	$\mathbf{22.4}_{\pm 0.1}$	$\mathbf{73.7}_{\pm 1.7}$	$\mathbf{369.6}_{\pm 4.6}$

Table 4.4: Results on the YouCook2 dataset [171]. † Our implementation on this dataset. sc: trained from scratch on YouCook2. pt: pretrained on the HowTo100M dataset, then finetuned on YouCook2.

	Text ⟹ Video				
Method	R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
Random	0.03	0.15	0.3	1675	1675
Ours (sc)	$16.6_{\pm 0.2}$	$37.4_{\pm 0.3}$	$48.3_{\pm 0.1}$	$12.0_{\pm 0.0}$	$95.5_{\pm 3.4}$
HT (pt) [104]	8.2	24.5	35.3	24	-
COOT (sc) [54]	$16.7_{\pm 0.4}$	$40.2_{\pm 0.3}$	$52.3_{\pm 0.5}$	$9.0_{\pm 0.0}$	-
MMT (pt) [50]†	$17.2_{\pm 0.4}$	$39.5_{\pm 0.7}$	$51.0_{\pm 0.5}$	$10.0_{\pm 0.0}$	$68.2_{\pm 0.9}$
Ours (pt p=0.8)	$\mathbf{23.2}_{\pm 0.5}$	$\mathbf{48.0}_{\pm 0.7}$	$\mathbf{58.6}_{\pm 0.8}$	$\mathbf{6.0}_{\pm 0.0}$	$\mathbf{60.4}_{\pm 3.0}$

In Table 4.5, we compare MSR-VTT results in two different settings: Training from scratch on MSR-VTT (sc) or pretraining on HowTo100M then finetuning on MSR-VTT (pt). When training from scratch, our method has a small drop in performance, when compared to MMT [50]. This is likely due to our approach using only 3 modalities instead of 7. Our method's performance is also weaker than a recent approach SSB [123] that uses a modified version of MMT. In the HowTo100M pretraining setting however, our modality masking approach outperforms the standard pretraining used in MMT, even if only 12% of MSR-VTT annotations are related to speech. Our results are competitive wrt SSB. We also show qualitative results of our method on this dataset in Fig. 4.4. Note how we perform well in the examples shown in the top two rows – both the queries refer to the

contents of speech. In the second row, while the correct video is retrieved at rank 5, the other videos in the top 5 also describe school systems, demonstrating the difficulty of the dataset where often a caption may be equally relevant to a number of videos.

Table 4.5: Comparison to state of the art on the 1K-A split [90] of the MSR-VTT dataset [159]. sc: trained from scratch on MSR-VTT. pt: pretrained on the HowTo100M dataset, then finetuned on MSR-VTT.

	Text $\Longrightarrow$ Video				
Method	R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
Random	0.1	0.5	1.0	500.5	500.5
JSFusion (sc) [162]	10.2	31.2	43.2	13	-
HT (sc) [104]	12.1	35.0	48.0	12	-
CE (sc) [90]	$20.9_{\pm 1.2}$	$48.8_{\pm 0.6}$	$62.4_{\pm 0.8}$	$6.0_{\pm 0.0}$	$28.2_{\pm 0.8}$
MMT (sc) [50]	$24.6_{\pm 0.4}$	$54.0_{\pm 0.2}$	$67.1_{\pm 0.5}$	$4.0_{\pm 0.0}$	$26.7_{\pm 0.9}$
Ours (sc)	$22.5_{\pm 0.9}$	$53.2_{\pm 1.5}$	$67.1_{\pm 0.4}$	$4.7_{\pm 0.5}$	$25.8_{\pm 0.3}$
SSB (sc) [123]	27.4	56.3	67.7	3.0	-
HT (pt) [104]	14.9	40.2	52.8	9	-
Hero (pt) [87]	20.5	47.6	60.9	-	-
FiT (pt) [17]	24.1	-	63.9	5	-
MMT (pt) [50]	$26.6_{\pm 1.0}$	$57.1_{\pm 1.0}$	$69.6_{\pm 0.0}$	$24.0_{\pm 0.8}$	
SSB (pt) [123]	**30.1**	58.5	69.3	**3.0**	-
Ours (pt p=0.8)	$28.7_{\pm 0.7}$	$\mathbf{59.5}_{\pm 0.7}$	$\mathbf{70.3}_{\pm 0.7}$	$3.8_{\pm 0.2}$	$\mathbf{23.0}_{\pm 0.5}$

Results on ActivityNet are presented in Table 4.6. The annotators of this dataset were explicitly required to ignore the audio track when describing the videos, therefore focusing the descriptions towards the visual modality. Our multi-modal pretraining approach hence yields similar results to the previous state-of-the-art method (SSB [123]).

4.4 Conclusion

We present a new pretraining method for learning a multimodal video encoder. It consists of an alternating modality masking strategy, where we mask and predict a different modality at each batch using the other available modalities. We show that this allows us to effectively pretrain a video encoder to jointly process RGB, audio and ASR, even on unlabelled datasets without manually-generated captions. Our method produces competitive results on five downstream video retrieval benchmarks,

Table 4.6: Paragraph to video retrieval performance on the ActivityNet dataset [80]. sc: trained from scratch on ActivityNet. pt: pretrained on the HowTo100M dataset, then finetuned on ActivityNet.

	Text $\Longrightarrow$ Video				
Method	R@1↑	R@5↑	R@50↑	MdR↓	MnR↓
Random	0.02	0.1	1.02	2458.5	2458.5
FSE (sc) [166]	$18.2_{\pm 0.2}$	$44.8_{\pm 0.4}$	$89.1_{\pm 0.3}$	7	-
CE (sc) [90]	$18.2_{\pm 0.3}$	$47.7_{\pm 0.4}$	$6.0_{\pm 0.0}$	$23.1_{\pm 0.5}$	
HSE (sc) [166]	20.5	49.3	-	-	-
MMT (pt) [50]	$28.7_{\pm 0.2}$	$61.4_{\pm 0.2}$	$94.5_{\pm 0.0}$	$3.3_{\pm 0.5}$	$\mathbf{16.0}_{\pm 0.4}$
SSB (pt) [123]	**29.2**	61.6	**94.7**	**3.0**	-
Ours (pt p=0.8)	$29.0_{\pm 0.5}$	$\mathbf{61.7}_{\pm 0.3}$	$94.6_{\pm 0.2}$	$4.0_{\pm 0.0}$	$16.8_{\pm 0.5}$

and is particularly suitable when user queries relate to the spoken language in videos.

Chapter 5

Improving speech recognition using vision

Automatic Speech Recognition (ASR) is often applied to edited or streamed media (for example, TV, online videos, video conferencing), where the input signal consists of both an audio and a visual stream. For these applications, the visual stream can provide strong cues for improving ASR, particularly in cases where the audio is degraded or corrupted. This has been largely exploited by AV-ASR works which focus on lip motion [2, 34, 94, 95, 115, 124, 142, 151] (using video crops centered around the speaker's mouth). While lip motion is a strong signal in videos centered on the speaker, it may be less useful in some online videos (those with egocentric viewpoints, face coverings, poor video quality, speaker at a distance etc.). A more recent and less-explored direction is the contribution of additional visual context, for example, the hand movements of a speaker, the presence of certain objects that are being described or even the background location [100].

In this chapter, we focus on the latter case. The main existing benchmark for this task is the How2 dataset, which consists of instructional videos where the ground truth is obtained from user uploaded transcripts [134]. While extremely valuable, the How2 dataset was created by keeping the audio samples that were most aligned to user-uploaded transcripts. This was done using an automatic alignment tool, and biases the dataset towards 'clean' audio samples. We posit that in this case, a model trained on clean audio would never be incentivised to learn from the visual modality, as all the information for ASR is present in the audio stream. This has led to a number of AV-ASR works to doubt whether the visual modality is useful at all in a clean audio context, or if it is simply used as a regularizer [23, 52, 145].

To resolve this issue, we explore masking strategies that degrade the audio samples during training, and then evaluate our model under noisy audio conditions. By dropping out key words in the

audio signal, we incentivise our model to pay attention to the visual stream. Our model is based on a Seq2Seq encoder-decoder architecture. Unlike previous works that use full-frame pre-extracted visual features [23, 52, 61, 100, 109, 116, 118, 134, 146], our encoder performs audio-visual fusion early, and is trained directly from pixels and spectrograms. We show that large performance gains can also be achieved by pretraining our model on the large HowTo100M [104] dataset (not to be confused with the How2 dataset).

Given the clean audio on the How2 test set, we also simulate noise at test time [52]. Unlike [52] which explicitly drops 'visual words', we simulate noise in a more objective way, by randomly dropping audio segments, or adding external environmental sounds from the AudioSet dataset [51]. We show that under these conditions, our model with audio-visual inputs consistently outperforms an audio only model for the task of AV-ASR. In addition, we also create a small, *real-world* test set with naturally occurring noisy audio. This dataset is created by filtering out samples where automatic ASR gets perfect results, and hence is a challenging test bed. On this dataset, we show that the visual modality makes a significant contribution to performance.

In this chapter we make the following contributions: (i) We propose a new encoder-decoder AudioVisual ASR TrAnsformeR (AVATAR) which is trained end-to-end from spectrograms and full-frame RGB. Our encoder fuses audio and visual inputs and is trained jointly with the decoder; (ii) To prevent the audio stream from dominating training, we propose and compare a number of masking strategies during training, thereby encouraging our model to pay attention to the visual stream; (iii) Our model achieves state-of-the-art performance on the How2 AV-ASR benchmark, with visual information improving performance under various simulated noise conditions; and finally (iv) we create a new, challenging *real-world* test bed for AV-ASR called VisSpeech. The dataset is created using a combination of automatic techniques and manual annotation and unlike other works, allows us to demonstrate the performance of our method under realistic noise conditions. We have released this dataset publicly to the research community at `https://gabeur.github.io/avatar-visspeech`.

5.1 Related Work

Audio-visual speech recognition. CTC [57] and Seq2Seq [33, 149] are the two most popular losses for performing ASR. In the context of AV-ASR focused on lip motion, they have been compared [2] and combined [94].

While early approaches [115, 151] use pre-extracted lip visual features, recent works [2, 34, 94, 95, 124, 142] adopt an end-to-end approach by directly processing the pixels of the speaker's lips. In contrast, the contribution of full frames for AV-ASR beyond the speaker's mouth movements has only been studied through pre-extracted visual context features: either action features [23, 52, 118, 134], place features [23, 61, 100, 116, 146] or object features [23, 61, 100, 109, 116, 146]. Unlike these works, we train directly from full frame pixels for AV-ASR.

Full-frame AV-ASR datasets. The How2 dataset [134], built from instructional videos, is the main benchmark for the full-frame AV-ASR task. Prior to this benchmark, full-frame AV-ASR works [61, 100, 116] have also evaluated on instructional videos datasets but those have not been released. Audio-captioned image datasets [145, 146] have also been used for AV-ASR. User-uploaded transcripts are the main source of ground truth for large scale AV-ASR video datasets [95, 134, 142]. As these are often misaligned or inaccurate, transcripts are typically first audio-aligned and then automatically filtered [88].

Audio signal degradation for AV-ASR evaluation. In the case of lip motion for AV-ASR, several works have experimented with adding 'babble noise' [2, 95] or extra speech tracks [95, 142]. Additive Gaussian noise has also been used in [142]. In the case of full-frame AV-ASR, it is common to completely mask some segments of the audio signal that correspond to visual words. Srinivasan et al. [146] mask the audio segments corresponding to nouns and places. Ghorbani et al. [52] compute the similarity between words and visual frames and mask only the visual words. We instead attempt to mimic more realistic settings by simulating audio degradations on How2 and releasing our 'in the wild' benchmark VisSpeech. In order to prevent audio from dominating the task, we extend audio degradation to the training phase by randomly masking input words at training time.

5.2 Methodology

5.2.1 Model Architecture

In this section we provide an overview of our audiovisual ASR model called AVATAR (Figure 5.1). Our model consists of a multimodal encoder to encode both RGB frames and audio spectrograms, and a transformer decoder which produces the natural language speech recognition output.

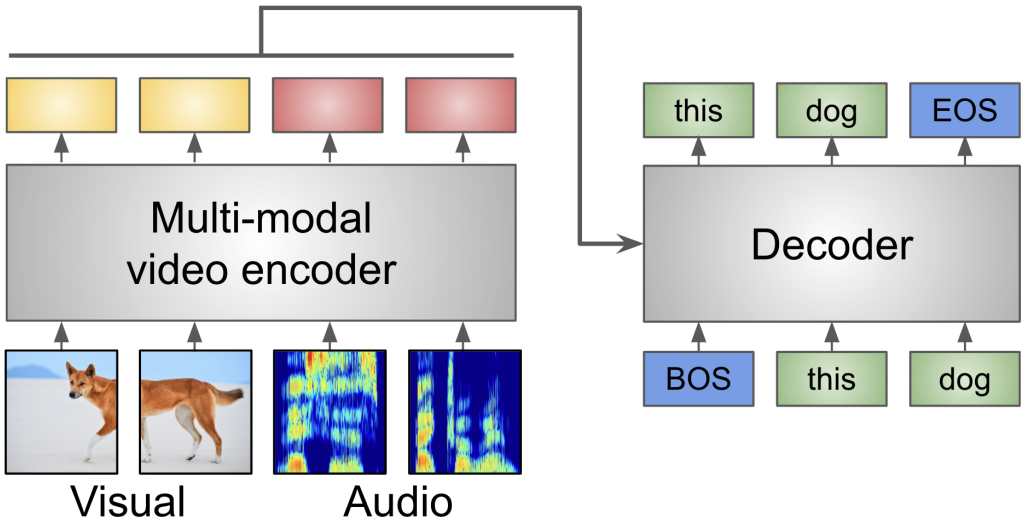

Figure 5.1: **AVATAR:** We propose a Seq2Seq architecture for audio-visual speech recognition. Our model is trained end-to-end from RGB pixels and spectrograms.

Unlike previous AV-ASR works, we do not use frozen visual features, but have a single multimodal encoder that allows early multimodal fusion [112].

Audio Inputs. Our model operates on 25 second audio inputs. We follow common practice and extract 80-dimensional filter bank features from the 16kHz raw speech signal using a Hamming window of 25ms and a stride of 10ms, giving us 80×2500 size spectrograms for 25 seconds of audio. We then extract 16×16 non overlapping patches, giving us a total of $5 \times 156 = 780$ input tokens for audio.
RGB Inputs. We randomly extract 2 frames at 2.5 fps from each input video clip, which are then converted into tokens by extracting $16 \times 16 \times 2$ tubelets resulting in a total of $14 \times 14 = 146$ input tokens (the image resolution is 224×224). This is based on our observations that visual signals are highly redundant for most videos and can therefore be efficiently captured from few frame samples.

Audiovisual Encoder. We adopt the recently proposed MBT architecture [112], which is a transformer based multimodal encoder. Given both sets of audio and RGB tokens, MBT first adds positional encodings to each token, and append a CLS token to each set. The sets are then fed to the MBT encoder. MBT relies on bottleneck tokens to model the

cross-modal interactions. Here we use the best parameters from [112] (4 bottleneck tokens and bottleneck fusion starting at layer 8). We use public ViT-Base [79] weights (ViT-B, $d_{model} = 768$ $L = 12$, $N_H = 12$, $d = 3072$)[1] pretrained on ImageNet-21K [39] for initialization.
Decoder. All hidden units from the encoder are then passed to an auto-regressive transformer decoder [153] consisting of 8 layers and 4 attention heads.

5.2.2 Training Strategies

In this section we describe our training loss and strategy for AVATAR. Our model is first pretrained on a large dataset with transcripts obtained using an *audio-only* ASR API [104]. To entice the model to pay attention to the visual modality, we introduce a word masking strategy, which is described below.

End to end training. The model is trained end-to-end using a cross-entropy loss on each decoded token.

Word Masking. To prevent our model from ignoring the visual modality, we introduce word masking techniques during training. We randomly sample target words and mask out the input audio signals that correspond to those words using pre-extracted alignments between words and the input signal. We obtain the alignment either from ASR results for pretraining or by using an off-the-shelf forced-alignment tool for finetuning. For selecting target words to mask, we experiment with two strategies: random and content word masking. The former strategy selects target words randomly whereas the latter chooses the targets among non-stop (henceforth known as content) words. For random word masking, we randomly mask out 10% of the words. For content word masking, there are fewer candidate words to be masked so content words are masked at a higher rate to match the 10% overall masking on the entire dataset.

5.3 VisSpeech Dataset

In this section we describe our new AV-ASR benchmark called VisSpeech. Our dataset is a subset of the publicly released HowTo100M dataset [104],

[1] d_{model} is the embedding dimension, L is the number of transformer layers, N_H is the number of self-attention heads with hidden dimension d.

and is curated using a combination of automatic filtering stages and manual verification.

Dataset creation pipeline:
Our dataset creation pipeline is driven by two objectives: (1) We want to find challenging audio conditions in which regular audio-based ASR fails. For this we seek videos where there is a large word error rate between automatic ASR and user-generated transcripts. (2) We are also interested in video segments where there is *high audio-visual correspondence*, in order to create a suitable multimodal test set for AV-ASR. Our pipeline consists of the following steps:

Step 1: Obtain videos with user uploaded transcripts. We first search for HowTo100M videos that have both manually uploaded transcripts and automatic ASR. We then align the transcripts using the Levenshtein algorithm in order to compute the global word error rate (WER) between the two. Videos with a global WER greater than 100% are removed, as we find this helps filter out completely wrong user uploaded transcripts or ASR. This gives us 85K videos.

Step 2: ASR vs user transcripts. Videos are split into segments, at silences detected using an open-source VAD model [152]. Using the user transcripts and the ASR from each segment, we filter out segments with WER greater than 50% between the two transcripts to remove samples with significantly low quality user transcripts or ASR. This is similar to the rationale for the filtering in Step 1, which is performed at a video level, but now we perform it at a segment level. Next and most importantly we remove “easy" samples, namely those with low WER of less than 20% for non-stop words (too clean) and samples with less than 9 words (too short). This leaves us with 773K sentences from 7.5K videos.

Step 3: Visual-Text similarity. To measure visual-text similarity, we run a video-text similarity model [110] trained on the Howto100M dataset to get similarity scores between each video and sentence pair. This helps highlight challenging samples where the visual modality can compensate for corrupted audio.

Step 4: Manual annotation. Finally, we manually check the highest similarity segments and correct the user-uploaded ASR if necessary.

VisSpeech consists of 508 segments from 495 unique videos. The average transcript length is 12.2 words and average segment duration is 4.3 seconds. By filtering for segments where ASR fails, we find that our dataset is truly a challenging test bed 'in the wild', with the audio containing background chatter, laughter, music and other environmental sounds. During the manual verification phase, we also noticed that many examples contain speech spoken with challenging English accents from various regions all over the world.

5.4 Experiments

In this section we first describe the datasets, metrics and implementation details for training (Section 5.4.1). We then describe the simulated noise we use for evaluating our models on the How2 dataset (Section 5.4.2). Finally we discuss the results of our model under different training strategies on both the How2 and our newly introduced VisSpeech dataset (Section 5.4.3).

5.4.1 Datasets, Metrics, Implementation Details

HowTo100M [104] consists of more than 1 million instructional videos associated with their automatically-extracted speech transcriptions. We only use this dataset for pretraining our model. Note that here we are not training with perfect ground truth, but using the ASR outputs of an existing model. We remove videos present in the validation and test sets of VisSpeech and How2 datasets (described next).
How2 [134] is an instructional video dataset created for multi-modal language understanding. We use the 300 hours version. The videos are segmented into short clips (avg 5.8s), each accompanied by their user-uploaded transcript (avg 20 words). The dataset is split between a training (184,949 clips), validation (2,022 clips) and test (2,305 clips).

Metrics. We evaluate our models using Word Error Rate (WER). For each sentence, dynamic programming is used to align the predicted words to the ground truth. The number of word errors (deletions, substitutions and insertions) is then computed across the whole test dataset and divided by the number of ground truth words to obtain the WER.

Training Implementation details. All models are trained end-to-end unless otherwise specified. We use a batch size of 1,536 and 256 for

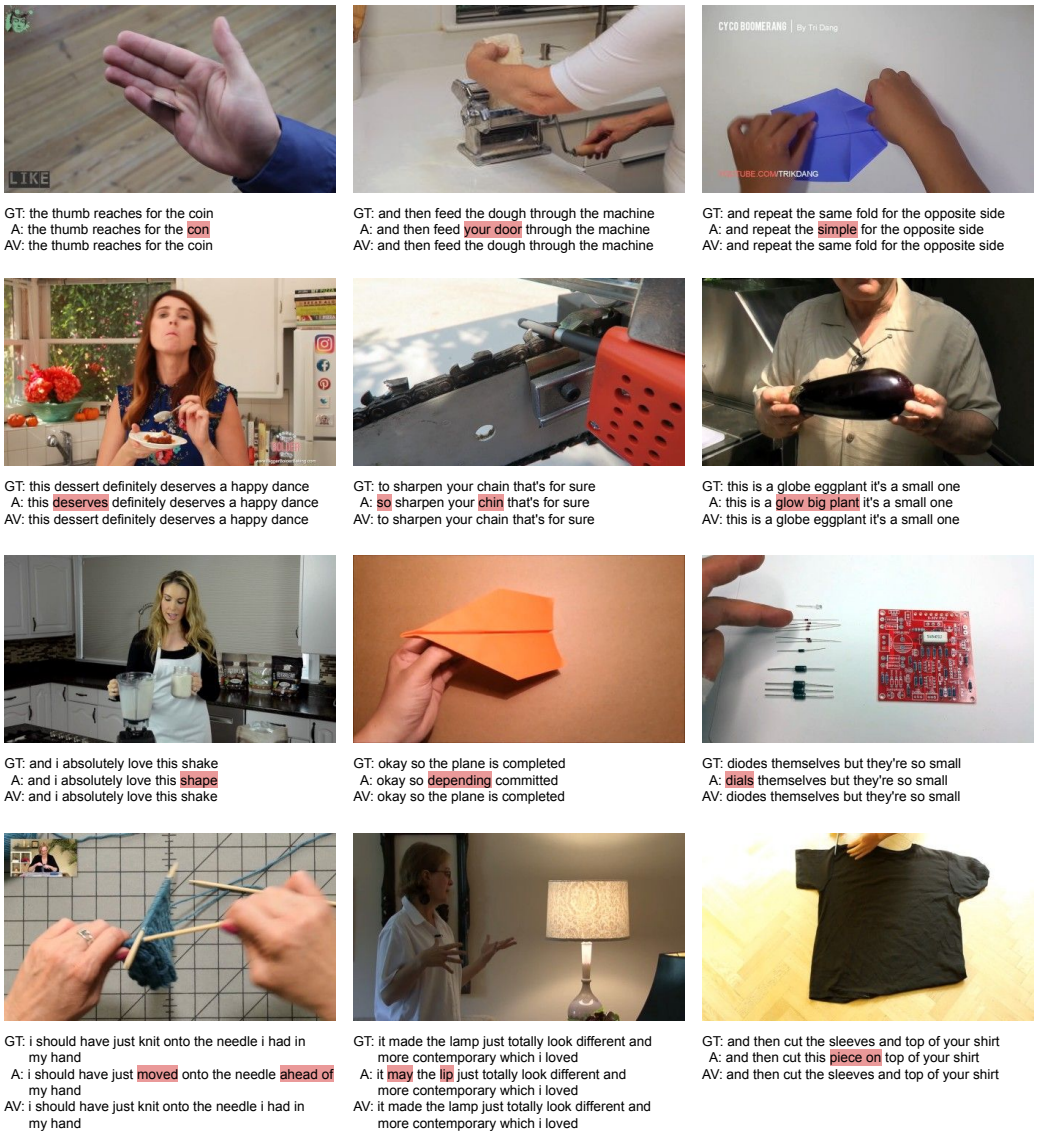

Figure 5.2: **Qualitative results on the VisSpeech dataset.** We show the ground truth (GT), and predictions from our audio only (A) and audio-visual model (A+V). Note how the visual context helps with objects ('chain', 'eggplant', 'coin', 'dough'), as well as actions ('knit', 'fold') which may be ambiguous from the audio stream alone. Errors in the predictions compared to the GT are highlighted in red.

pretraining and finetuning respectively. We adopt a wordpiece tokenizer [155] pretrained for BERT and decode using a beam search with a beam size of 4 and a brevity penalty of 0.6. We use SpecAugment with parameters adopted from [119]. We augment the visual frames using random cropping and color jittering. We use a momentum optimizer. We pretrain our model for 1M iterations with an initial learning rate of 2. The learning rate is warmed up for 1K iterations and then linearly decayed to 0. We initialize both visual and audio streams of the MBT encoder with the public ViT [79] weights pretrained on ImageNet. For finetuning, we train for 40K iterations without warmup.

5.4.2 Simulated Noise for Evaluation

The ground-truth transcripts in How2 are collected by performing forced-alignment on the user uploaded transcripts, and filtering out examples with a low confidence score. Due to such a filtering process, the audio signals in How2 are inherently clean, and consequently the task is largely audio dominant. To overcome this limitation, we evaluate our models with three types of simulated audio noise: burst packet loss, environment noise and mixed noise. For the burst packet loss, we randomly drop two chunks of the input audio signal where the length of each chunk is uniformly sampled from $(0, 0.1]$ times the video duration. To simulate environment noise, we add audio noise randomly sampled from the 'noise' and 'environmental' classes in the AudioSet dataset [51]. Finally, we also evaluate a combination of the two ('mixed noise'). Note that we train a single model and evaluate it under different noise configurations.

5.4.3 Results

Effects of Training Strategies. Table 5.2 compares audio-only (A) and audiovisual (A+V) models trained with different training strategies and evaluation noise settings on How2. In addition to clean audio, we report results in three degraded audio scenarios, burst loss, environment noise and mixed noise. We first note that without pretraining, our model performs well on the clean eval, but performance degrades significantly under simulated noise conditions. Adding the visual modality under these noise conditions helps performance across the board. Vanilla pretraining then improves performance significantly, however we note the gap between A and A+V also shrinks, signifying the improvement

is largely from better audio encoding. In this case, the A+V model has no incentive to look at visual inputs, as the task under clean conditions is dominated by audio. In addition, the pretraining is performed on HowTo100M where the transcripts are automatically generated from the audio alone, and so our model is able to solve the task without any visual information. We find the word masking strategies to be extremely effective to mitigate this. The overall performance of both A and A+V are improved and notably, the improvements of A+V are larger. We show that with these masking schemes, adding visual inputs helps even for clean audio, with the performance improving under environment and mixed noise. Note that across the board, adding the visual modality improves performance.

Comparison to the state-of-the-art. Table 5.1 compares AVATAR with existing state-of-the-art methods on How2. Our model trained from scratch already outperforms all existing methods and serves as a strong baseline. Our best model, which is pretrained on HowTo100M, with the random word masking technique, brings a further boost reducing the error rate by over 45% relatively compared to the existing state-of-the-art method.

Table 5.1: **Comparison to the state-of-the-art on How2.** Our model outperforms all previous works when trained from scratch, and pretraining provides a significant boost. We report the best audio-visual numbers for all works.

Model	**%WER**
BAS [134]	18.0
VAT [23]	18.0
MultiRes [118]	20.5
LLD [52]	16.7
AVATAR (scratch)	15.6
AVATAR (pretrained)	**9.1**

Table 5.2: Audiovisual ASR vs Audio only models under various *evaluation* noise conditions (Clean, Burst, Environment and Mixed) and with different *training* masking strategies (Random and Content). Percentage Word Error Rate (%WER) is reported on the How2 test set. **A:** Audio-only. **A+V:** Audiovisual. **Rel. Δ:** Relative improvement of A+V over A.

Training \ Eval Noise	Clean			Burst Loss			Environment Noise			Mixed Noise		
	A	**A+V**	**Rel. Δ**	**A**	**A+V**	**Rel. Δ**	**A**	**A+V**	**Rel. Δ**	**A**	**A+V**	**Rel. Δ**
No Pretraining	15.72	15.62	0.64%	29.59	28.69	3.05%	50.79	47.70	6.08%	60.51	57.49	5.0%
Vanilla Pretraining	9.75	9.79	-0.33%	21.97	21.71	1.19%	25.97	25.55	1.61%	39.13	38.96	0.42%
Random Word Masking	9.19	9.11	0.93%	15.60	15.28	2.05%	23.39	22.35	4.45%	32.43	30.64	5.50%
Content Word Masking	9.58	9.25	3.48%	17.26	16.92	1.98%	23.77	22.67	4.65%	33.83	32.26	4.53%

Evaluation on VisSpeech. We evaluate our AVATAR model trained with different strategies on VisSpeech with real-world noise (Table 5.3). We finetune the pretrained models for 5K iterations on How2. Our dataset effectively highlights the contribution of the visual modality without introducing any artificial noise. Once again, both masking strategies help the audiovisual (A+V) model learn to utilize the visual modality better. Content word masking improves the performance only when the visual modality is provided; providing some evidence that the A+V model uses visual inputs to correct errors on content words. To further tease apart the input of the visual modality, we compute the word error rate on content words *only* and on stop words *only*. This is because we hypothesize that visual modality should not be able to provide any useful information about stop words, and so most of the improvement should be on the content words. As expected, we find the errors on content words are reduced substantially more than those on stop words when the visual modality is incorporated with all training strategies (e.g., 1.18% vs. 0.33% absolute error rate drops with content word masking). This also confirms the contribution of the visual modality. Further evidence can be found from the qualitative examples provided in Figure 5.2, where it can be clearly seen that visual context helps with correcting ASR errors on objects as well as actions.

Table 5.3: WERs of AVATAR on our newly introduced test set VisSpeech consisting of real-world noise. The models are trained on automatic ASR from HowTo100M, and finetuned on How2. Note here we do not add any artificial audio degradation at all.

Training Strategy	**A**	**A+V**	**Rel. Δ**
No pretraining	44.57	43.41	2.61%
Vanilla	12.69	11.91	6.11%
Random Word Masking	12.35	11.86	3.93%
Content Word Masking	12.72	11.28	11.30%

End-to-end training with early audiovisual fusion. Unlike previous works on full-frame AV-ASR, AVATAR is trained (i) entirely end-to-end, and (ii) with early audiovisual fusion in the MBT encoder. To assess this effect, we test AVATAR with pre-extracted visual features as in [52], using a model pretrained on HowTo100M with NCE loss [110]. We concatenate the audio features at the output of our MBT encoder with our pre-extracted

visual features and provide them to the decoder. Note that in this case the audio-visual fusion happens only through the decoder. We train this model with random masking strategy and find that the end-to-end trained model outperforms the model with pre-extracted features with 9% relative improvement in the mixed noise setting and similar trends are observed in all the other noise settings.

Contribution of Visual Modality. Some works show that the visual modality is simply a regularizer [23, 52, 145]. As done by [145], we further investigate whether the contribution of the visual modality is simply a regularizer by replacing the visual frames of test examples with those extracted from random validation videos. Unlike [145], we observe significant degradation of A+V models in all settings (e.g., 9.11%→9.53% with random word masking and clean audio) as the models get distracted by the random visual inputs. Note that this performance (9.53%) is worse than the audio-only model in this setting (9.19%). This suggests vision in our model is not simply a regularizer contrary to what was previously reported in [23, 52].

5.5 Conclusion

In this chapter, we proposed a novel encoder-decoder transformer architecture and training strategies based on word masking for AV-ASR. We showed that our method helps the model learn to use visual inputs better and outperform the state of the art. Finally we also presented VisSpeech, a new AV-ASR test benchmark, and demonstrated the effectiveness of our method under naturally occurring noise.

Chapter 6

Conclusion

In this thesis, we studied the problem of multi-modal video understanding by focusing on two tasks: text-to-video retrieval and audio-visual speech recognition. We showed that video media semantics are scattered across multiple modalities that should be processed jointly in order to avoid discarding valuable cross-modal information. We designed and trained deep learning models capable of extracting such cross-modal cues. In this last chapter, we summarize the contributions of the thesis in Section 6.1 and then conclude with perspectives for future research in Section 6.2.

6.1 Summary of Contributions

Modality fusion with the multi-modal transformer. In Chapter 3, we tackled the task of caption-to-video retrieval. Having noted that video media presents both cross-modal and temporal information, we proposed a transformer-based model capable of extracting video semantics across these two dimensions. Our multi-modal transformer takes as input sequences of pre-computed features extracted by seven expert models trained on specialized datasets to recognize motion, scene, objects, speech, faces, text and sounds. By leveraging the self-attention mechanism, our multi-modal transformer lets all features collaborate with each other to extract both cross-modal and temporal semantics from the video. We also investigated several approaches to encode text captions and obtained the best results with a BERT pretrained model. We integrated both our multi-modal transformer and caption encoder into a cross-modal architecture and obtained state-of-the-art results at the time of publication on the MSRVTT, ActivityNet and LSMDC datasets.

Multi-modal pretraining of a video encoder. Training the cross-modal architecture introduced in Chapter 3 requires a large dataset of video-caption pairs which can be expensive and difficult to obtain. In order to leverage the large amount of unlabeled videos available on the internet, self-supervised techniques have been proposed to pretrain the encoders using speech as pseudo-captions. However, the video encoder is in this case deprived of the speech modality and therefore never pretrained on it. In Chapter 4, we proposed a self-supervised approach to pretrain the video encoder on three video modalities: appearance, audio and speech. At each training batch, we randomly select one of these modalities and completely mask it from the video encoder. It is instead processed by the query encoder to serve as supervision to video encoder training. This results in the video encoder being pretrained on all the video modalities, including speech. Our modality masking pretraining approach obtains state-of-the-art results on three speech-rich video datasets at the time of publication: CMD, YouCook2, and How2R.

Visual context for improved speech recognition. As shown in Chapter 4, speech can be a crucial modality to correctly understand a video, it therefore needs to be properly extracted from the audio-visual signal. In Chapter 5 we proposed an encoder-decoder architecture to tackle this task. Our model, called AVATAR, takes audio spectrograms and pixels as input and outputs a sequence of tokens as the predicted transcript. We showed that standard pretraining on the HowTo100M dataset provides a large performance improvement that is mainly due to better audio processing but not better vision processing. In order to entice the model to use the visual modality, we presented two word-masking techniques: random word masking and content word masking. We introduced a challenging audio-visual speech recognition benchmark called VisSpeech on which content word masking provides a large performance boost to speech recognition and helps our model better leverage the visual modality.

6.2 Perspectives for future research

6.2.1 Video speech estimation training for visual-language tasks

Large language models (LLMs) trained auto-regressively on vast amounts of textual data [21] have recently obtained impressive zero shot performance on a variety of NLP tasks including question answering and text generation. These pure NLP models are however not directly suitable for vision processing. Words like 'chair' or 'apple' are generated without a clue of how those objects look like in the real world. This prevents the extension of their question answering aptitudes to visual tasks like visual question answering or video captioning.

One potential research direction is to leverage video media to auto-regressively train a visual-language model on spoken language estimation. The model would then be able to respond to visual questions by generating natural language and tackle tasks like video captioning, video classification or visual question answering.

In a very recent work called Flamingo [6], Alayrac et al. extend LLMs with visual processing capability. Cross-attention layers are added to a frozen language model to attend visual features and the whole model is trained auto-regressively to generate the text of large scale visual-captions datasets. A mixture of three kinds of datasets is used to train Flamingo: interleaved images and text extracted from webpages, image-text pairs and video-text pairs, mainly obtained through alt-text descriptions. As presented in figure 6.1, one of the failure cases of the approach reported by the authors is hallucinations.

Focusing on the spoken language accompanying videos instead of alt-text or manually generated captions could help with these shortcomings. Contrary to captions, uncertainty is commonly expressed in video speech, which is a desirable feature to tackle the failure cases presented in figure 6.1. In response to the question "What is it?", a video character could reply "I don't know", "I can't say", or "I am not sure but I think ...". A visual-language model trained on such data could learn to reply with uncertainty to tricky questions.

Interacting with a model to solve visual-language tasks trough natural language prompting naturally takes the form of a dialogue, with questions from the user being followed by responses from the model, e.g. "Who

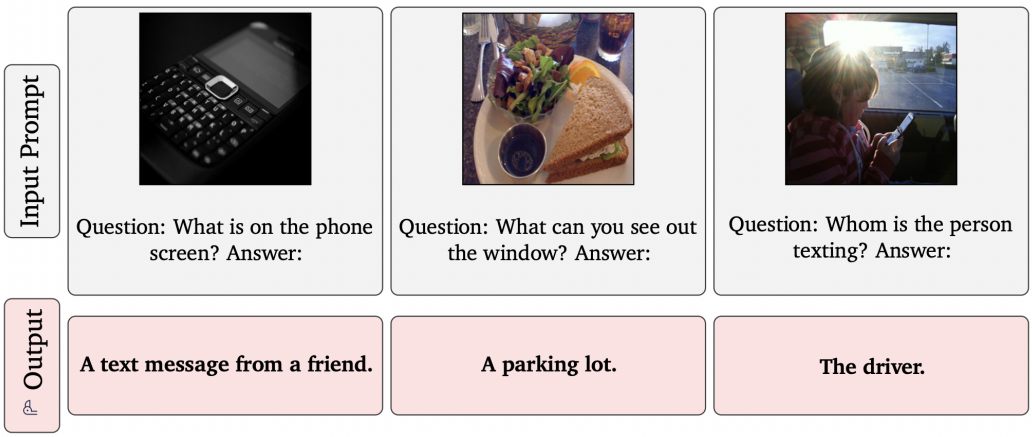

Figure 6.1: Some failure cases of Flamingo [6]. When prompted with tricky questions, the model sometimes hallucinates or takes ungrounded guesses. (Image courtesy of Alayrac et al. [6]).

painted this? Paul Gauguin. When? 1892.". Dialogue is often present in video speech but is usually absent from captions. Training on video speech could hence reduce the domain gap between training and inference data. Spoken language in videos is also temporally aligned to the visual modality and could be leveraged to automatically segment and label a video, e.g. [1:10->1:45] "This is a giraffe", [1:45->2:30] "This is a lion".

However, finetuning a LLM to process whole visual frames by auto-regressively predicting speech also comes with a few challenges.

The first one is how to bridge the domain gap in transcription between spoken language and written language. Spoken language transcriptions would probably have to be made closer to written language using punctuation and removal of hesitations (self-corrections and repeated words or partial words that appear in speech). Additional transformation of the training video transcripts could be required to change first-person narrations such as "Now what will we do? We'll paint it!" to third-person narrations hence matching the third-person dialogue prompting "Now what will they do?" used at inference. This transformation of the transcripts could be performed automatically similar to [161] that generate questions and answers from speech transcripts.

Another challenge would be to prevent the model from focusing on the task of lip reading instead of attending the whole visual frame and learning complex visual-language interactions. As explained in Chapter 5,

this would apply to videos where the speaker is visible, in high definition and with high framerate. Reducing the framerate (to 1fps for examples) could therefore be a straightforward solution to prevent lip-reading. Going further, if we wanted the model to not only process the visual signal but also the audio signal of a video, we would need to prevent the model from only learning to perform the AV-ASR task presented in Chapter 5. This could be done similar to [140] by introducing some temporal misalignment between the video and the transcript to be predicted, e.g., predicting future speech utterances.

6.2.2 Speech recognition in long duration videos

In Chapter 5, we have shown how the visual context can help better transcribe what is being said in a video clip. However, similar to most previous speech recognition works, we trained and evaluated our model on artificially short utterances. Indeed, commonly used datasets for ASR (Librispeech [117], Switchboard [55]) or AV-ASR (LRS3-TED [3], How2 [134]) are constituted of short segments extracted from longer audio sequences or video clips. Evaluating on such short utterances puts an artificial focus on the intra-utterance context and discard the long-range context that could be leveraged if attending to the whole signal duration.

One potential research direction is to perform speech recognition on full-length video instead of selected segments. Long-range context (both audio and visual) can provide strong cues for correctly transcribing a video (Fig. 6.2). What has been said earlier or what will be said later in a video can provide precious context information (e.g., the subject of the conversation) when trying to recognize the speech in a specific segment. As shown in [102], the visual context necessary to correctly transcribe a video segment might not be synchronized with speech and could therefore also require to attend a longer range than the immediate context. The limited utterance duration of commonly used datasets of ASR and AV-ASR has not allowed a fair comparison between the contributions of spoken context versus visual context.

There are two main challenges that explain why the community has focused on short crops instead of long sequences.

First, ground truth transcripts for long utterances are hard to obtain.

Figure 6.2: Long range context might be required to correctly transcribe a word. In the case of a movie for example, the name of a character might be pronounced early in the movie but be impossible for the model to correctly transcribe using immediate context. It is only with the film credits at the very end of the movie that the correct transcription can be extracted from the visual modality.

Manually annotating spoken language in a video can be very difficult, especially in noisy conditions, in presence of strong accents or regional dialects. It also often requires expert knowledge about the subject being discussed. Ground truth transcripts are therefore usually obtained by aligning a user-provided transcript and selecting 'islands of confidence' where the audio aligns well with the provided transcript [88].

Another challenge is that audio (and video) are very high dimensional but processing capacity is limited. The ability to recognise subtle variations in the audio signal is crucial for speech recognition, but also computationally expensive. It is currently not practical to provide an hour-long audio (or video) signal to be processed for speech recognition.

Tackling the first one of these challenges would require a large effort of manual annotation but is technically feasible. Movie transcripts could constitute a good starting point before manual annotation refinement.

Several efficient transformer architectures [20, 72] have been proposed in response to the second challenge but the context length is increased at

the expense of accuracy. A possible approach to the problem would be to decompose speech prediction in two steps. A first step could focus on extracting short range features in several modalities that could then be leveraged by a second step to specifically attend the video segments that are relevant to the speech recognition of a specific segment.

Bibliography

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Apostol (Paul) Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. YouTube-8M: A large-scale video classification benchmark. In *arXiv:1609.08675*, 2016.

[2] Triantafyllos Afouras, Joon Son Chung, Andrew W. Senior, Oriol Vinyals, and Andrew Zisserman. Deep audio-visual speech recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[3] Triantafyllos Afouras, Joon Son Chung, and Andrew Zisserman. Lrs3-ted: a large-scale dataset for visual speech recognition. *ArXiv*, abs/1809.00496, 2018.

[4] Hassan Akbari, Liangzhe Yuan, Rui Qian, Wei-Hong Chuang, Shih-Fu Chang, Yin Cui, and Boqing Gong. VATT: transformers for multimodal self-supervised learning from raw video, audio and text. *CoRR*, abs/2104.11178, 2021.

[5] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4575–4583, 2016.

[6] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. *ArXiv*, abs/2204.14198, 2022.

[7] Jean-Baptiste Alayrac, Adrià Recasens, Rosalia Schneider, Relja Arandjelović, Jason Ramapuram, Jeffrey De Fauw, Lucas Smaira, Sander Dieleman, and Andrew Zisserman. Self-supervised multimodal versatile networks. In *NeurIPS*, 2020.

[8] Samuel Albanie, Yang Liu, Arsha Nagrani, Antoine Miech, Ernesto Coto, Ivan Laptev, Rahul Sukthankar, Bernard Ghanem, Andrew Zisserman, Valentin Gabeur, Chen Sun, Alahari Karteek, Cordelia Schmid, Shizhe Chen, Yida Zhao, Qin Jin, Kaixu Cui, Hui Liu, Chen Wang, Yudong Jiang, and Xiaoshuai Hao. The end-of-end-to-end: A video understanding pentathlon challenge (2020). *ArXiv*, abs/2008.00744, 2020.

[9] Relja Arandjelović, Petr Gronát, Akihiko Torii, Tomás Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5297–5307, 2016.

[10] Relja Arandjelović and Andrew Zisserman. Look, listen and learn. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 609–617, 2017.

[11] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lucic, and Cordelia Schmid. Vivit: A video vision transformer. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6816–6826, 2021.

[12] Yuki M. Asano, Mandela Patrick, Christian Rupprecht, and Andrea Vedaldi. Labelling unlabelled videos from scratch with multi-modal self-supervision. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[13] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016.

[14] Alexei Baevski, Henry Zhou, Abdel rahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *ArXiv*, abs/2006.11477, 2020.

[15] Parnia Bahar, Patrick Wilken, Tamer Alkhouli, Andreas Guta, Pavel Golik, Evgeny Matusov, and Christian Herold. Start-before-end and end-to-end: Neural speech translation by apptek and rwth aachen university. In *IWSLT*, 2020.

[16] Max Bain, Arsha Nagrani, Andrew Brown, and Andrew Zisserman. Condensed movies: Story based retrieval with contextual embeddings. In *ACCV*, 2020.

[17] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *ICCV*, 2021.

[18] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *ArXiv*, abs/2106.08254, 2021.

[19] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *ArXiv*, abs/2105.04906, 2021.

[20] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150, 2020.

[21] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *ArXiv*, abs/2005.14165, 2020.

[22] Andrea Burns, Reuben Tan, Kate Saenko, Stan Sclaroff, and Bryan A. Plummer. Language Features Matter: Effective language representations for vision-language tasks. In *ICCV*, 2019.

[23] Ozan Caglayan, Ramon Sanabria, Shruti Palaskar, Loïc Barrault, and Florian Metze. Multimodal grounding for sequence-to-sequence speech recognition. *ICASSP*, 2019.

[24] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018.

[25] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *ArXiv*, abs/2006.09882, 2020.

[26] J. Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *CVPR*, 2017.

[27] William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964, 2016.

[28] David L. Chen and William B. Dolan. Collecting highly parallel data for paraphrase evaluation. In *ACL*, 2011.

[29] Honglie Chen, Weidi Xie, Triantafyllos Afouras, Arsha Nagrani, Andrea Vedaldi, and Andrew Zisserman. Audio-visual synchronisation in the wild. In *BMVC*, 2021.

[30] Honglie Chen, Weidi Xie, Andrea Vedaldi, and Andrew Zisserman. Vggsound: A large-scale audio-visual dataset. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 721–725, 2020.

[31] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020.

[32] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15745–15753, 2021.

[33] 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 *EMNLP*, 2014.

[34] Joon Son Chung, Andrew W. Senior, Oriol Vinyals, and Andrew Zisserman. Lip reading sentences in the wild. *CVPR*, 2017.

[35] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[36] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[37] George V. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2:303–314, 1989.

[38] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[39] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[40] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[41] Carl Doersch, Abhinav Kumar Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. *2015 IEEE*

International Conference on Computer Vision (ICCV), pages 1422–1430, 2015.

[42] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021.

[43] Maksim Dzabraev, Maksim Kalashnikov, Stepan Komkov, and Aleksandr Petiushko. Mdmmt: Multidomain multimodal transformer for video retrieval. In *CVPR Workshop*, 2021.

[44] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6201–6210, 2019.

[45] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[46] Valentin Gabeur, Jean-Sebastien Franco, Xavier Martin, Cordelia Schmid, and Gregory Rogez. Moulding humans: Non-parametric 3d human shape estimation from single images. In *International Conference on Computer Vision (ICCV)*, 2019.

[47] Valentin Gabeur, Arsha Nagrani, Chen Sun, Karteek Alahari, and Cordelia Schmid. Masking modalities for cross-modal video retrieval. In *Winter Conference on Applications of Computer Vision (WACV)*, 2022.

[48] Valentin Gabeur, Paul Hongsuck Seo, Arsha Nagrani, Chen Sun, Karteek Alahari, and Cordelia Schmid. Avatar: Unconstrained audiovisual speech recognition. In *INTERSPEECH*, 2022.

[49] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. CVPR 2020 video pentathlon challenge: Multi-modal transformer for video retrieval. In *CVPR Video Pentathlon Workshop*, 2020.

[50] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal Transformer for Video Retrieval. In *ECCV*, 2020.

[51] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. *ICASSP*, 2017.

[52] Shahram Ghorbani, Yashesh Gaur, Yu Shi, and Jinyu Li. Listen, look and deliberate: Visual context-aware speech recognition using pre-trained text-video representations. In *IEEE Spoken Language Technology Workshop (SLT)*, 2021.

[53] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *ArXiv*, abs/1803.07728, 2018.

[54] Simon Ging, Mohammadreza Zolfaghari, Hamed Pirsiavash, and Thomas Brox. COOT: Cooperative hierarchical transformer for video-text representation learning. In *NeurIPS*, 2020.

[55] John J. Godfrey and Edward Holliman. Switchboard-1 Release 2 LDC97S62. *Linguistic Data Consortium*, 1993.

[56] Alex Graves. Sequence transduction with recurrent neural networks. *ArXiv*, abs/1211.3711, 2012.

[57] Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. *ICML*, 2006.

[58] Alex Graves, Abdel rahman Mohamed, and Geoffrey E. Hinton. Speech recognition with deep recurrent neural networks. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, 2013.

[59] Jean-Bastien Grill, Florian Strub, Florent Altch'e, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *ArXiv*, abs/2006.07733, 2020.

[60] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented transformer for speech recognition. *ArXiv*, abs/2005.08100, 2020.

[61] Abhinav Gupta, Yajie Miao, Leonardo Neves, and Florian Metze. Visual features for context-aware speech recognition. In *ICASSP*, 2017.

[62] D. Harwath, A. Recasens, D. Surís, G. Chuang, A. Torralba, and J. Glass. Jointly discovering visual objects and spoken words from raw sensory input. In *ECCV*, 2018.

[63] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735, 2020.

[64] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[65] Hermann von Helmholtz. Die lehre von den tonempfindungen als physiologische grundlage für die theorie der musik. *Die Lehre von den Tonempfindungen als physiologische Grundlage für die Theorie der Musik*, 1863.

[66] Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, R. Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, Malcolm Slaney, Ron J. Weiss, and Kevin W. Wilson. Cnn architectures for large-scale audio classification. *ICASSP*, 2017.

[67] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8), Nov. 1997.

[68] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2019.

[69] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2016.

[70] Gabriel Huang, Bo Pang, Zhenhai Zhu, Clara Rivera, and Radu Soricut. Multimodal pretraining for dense video captioning. In *AACL*, 2020.

[71] A. G. Ivakhnenko. Polynomial theory of complex systems. *IEEE Trans. Syst. Man Cybern.*, 1:364–378, 1971.

[72] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver: General perception with iterative attention. In *ICML*, 2021.

[73] Frederick Jelinek. Statistical methods for speech recognition. In *Statistical methods for speech recognition*, 1997.

[74] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:221–231, 2013.

[75] Andrej Karpathy, Armand Joulin, and Li Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. In *NeurIPS*, 2014.

[76] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[77] Henry J. Kelley. Gradient theory of optimal flight paths. *ARS Journal*, 30:947–954, 1960.

[78] Benjamin Klein, Guy Lev, Gil Sadeh, and Lior Wolf. Associating neural word embeddings with deep image representations using fisher vectors. In *CVPR*, 2015.

[79] Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[80] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *ICCV*, 2017.

[81] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[82] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *ACL*, 2018.

[83] Hang Le, Juan Miguel Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. Dual-decoder transformer for joint automatic speech recognition and multilingual speech translation. In *COLING*, 2020.

[84] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86:2278–2324, 1998.

[85] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. Stacked cross attention for image-text matching. *ECCV*, 2018.

[86] 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 *CVPR*, 2021.

[87] 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 *EMNLP*, 2020.

[88] Hank Liao, Erik McDermott, and Andrew Senior. Large scale deep neural network acoustic modeling with semi-supervised training data for youtube video transcription. In *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013.

[89] Qiubin Lin, Wenming Cao, and Zhiquan He. Level-wise aligned dual networks for text–video retrieval. *EURASIP Journal on Advances in Signal Processing*, 2022.

[90] Yang Liu, Samuel Albanie, Arsha Nagrani, and Andrew Zisserman. Use what you have: Video retrieval using representations from collaborative experts. In *BMVC*, 2019.

[91] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision, Kerkyra, Corfu, Greece, September 20-25, 1999*, pages 1150–1157. IEEE Computer Society, 1999.

[92] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.*, 60(2):91–110, 2004.

[93] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019.

[94] Pingchuan Ma, Stavros Petridis, and Maja Pantic. End-to-end audio-visual speech recognition with conformers. *ICASSP*, 2021.

[95] Takaki Makino, Hank Liao, Yannis M. Assael, Brendan Shillingford, Basi García, Otavio Braga, and Olivier Siohan. Recurrent neural network transducer for audio-visual speech recognition. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019.

[96] Jonathan Malmaud, Jonathan Huang, Vivek Rathod, Nick Johnston, Andrew Rabinovich, and Kevin P. Murphy. What's cookin'? interpreting cooking videos using text, speech and vision. In *NAACL*, 2015.

[97] Andrew McCallum and Kamal Nigam. A comparison of event models for naive bayes text classification. In *AAAI 1998*, 1998.

[98] Iain McCowan, David Dean, Mitchell McLaren, Robbie Vogt, and Sridha Sridharan. The delta-phase spectrum with application to voice activity detection and speaker recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 19:2026–2038, 2011.

[99] Martin F. McKinney and Jeroen Breebaart. Features for audio and music classification. In *ISMIR*, 2003.

[100] Yajie Miao and Florian Metze. Open-domain audio-visual speech recognition: A deep learning approach. In *INTERSPEECH*, 2016.

[101] Antoine Miech, Jean-Baptiste Alayrac, Ivan Laptev, Josef Sivic, and Andrew Zisserman. Thinking fast and slow: Efficient text-to-visual retrieval with transformers. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9821–9831, 2021.

[102] 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 *CVPR*, 2020.

[103] Antoine Miech, Ivan Laptev, and Josef Sivic. Learning a text-video embedding from incomplete and heterogeneous data. *ArXiv*, abs/1804.02516, 2018.

[104] 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 *ICCV*, 2019.

[105] Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *ICLR*, 2013.

[106] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, 2013.

[107] Niluthpol C Mithun, Juncheng Li, Florian Metze, and Amit K Roy-Chowdhury. Learning joint embedding with multimodal cues for cross-modal video-text retrieval. In *ICMR*, 2018.

[108] Niluthpol C Mithun, Juncheng Li, Florian Metze, and Amit K Roy-Chowdhury. Joint embeddings with multimodal cues for video-text retrieval. *IJMIR*, 2019.

[109] Yasufumi Moriya and G. Jones. Lstm language model adaptation with images and titles for multimedia automatic speech recognition. *IEEE Spoken Language Technology Workshop (SLT)*, 2018.

[110] Arsha Nagrani, Paul Hongsuck Seo, Bryan Seybold, Anja Hauth, Santiago Manen, Chen Sun, and Cordelia Schmid. Learning audio-video modalities from image captions. *ArXiv*, abs/2204.00679, 2022.

[111] Arsha Nagrani, Chen Sun, David Ross, Rahul Sukthankar, Cordelia Schmid, and Andrew Zisserman. Speech2action: Cross-modal supervision for action recognition. In *CVPR*, 2020.

[112] Arsha Nagrani, Shan Yang, Anurag Arnab, Aren Jansen, Cordelia Schmid, and Chen Sun. Attention bottlenecks for multimodal fusion. *NeurIPS*, 2021.

[113] Yoshitaka Nakajima, Mizuki Matsuda, Kazuo Ueda, and Gerard Remijn. Temporal resolution needed for auditory communication: Measurement with mosaic speech. *Frontiers in Human Neuroscience*, 12, 04 2018.

[114] Joe Yue-Hei Ng, Matthew J. Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4694–4702, 2015.

[115] Kuniaki Noda, Yuki Yamaguchi, Kazuhiro Nakadai, H. Okuno, and Tetsuya Ogata. Audio-visual speech recognition using deep learning. *Applied Intelligence*, 42:722–737, 2014.

[116] Shruti Palaskar, Ramon Sanabria, and Florian Metze. End-to-end multimodal speech recognition. *ICASSP*, 2018.

[117] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An asr corpus based on public domain audio books. *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015.

[118] Georgios Paraskevopoulos, Srinivas Parthasarathy, Aparna Khare, and Shiva Sundaram. Multimodal and multiresolution speech recognition with transformers. In *ACL*, July 2020.

[119] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin Dogus Cubuk, and Quoc V. Le. Specaugment: A simple data augmentation method for automatic speech recognition. In *INTERSPEECH*, 2019.

[120] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *BMVC*, 2015.

[121] Mandela Patrick, Yuki M. Asano, Bernie Huang, Ishan Misra, Florian Metze, João F. Henriques, and Andrea Vedaldi. Space-time crop & attend: Improving cross-modal video representation learning. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10540–10552, 2021.

[122] Mandela Patrick, Yuki M. Asano, Polina Kuznetsova, Ruth C. Fong, João F. Henriques, Geoffrey Zweig, and Andrea Vedaldi. On compositions of transformations in contrastive self-supervised learning. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9557–9567, 2021.

[123] Mandela Patrick, Po-Yao Huang, Yuki Asano, Florian Metze, Alexander G Hauptmann, Joao F. Henriques, and Andrea Vedaldi. Support-set bottlenecks for video-text representation learning. In *ICLR*, 2021.

[124] Stavros Petridis, Themos Stafylakis, Pingchuan Ma, Feipeng Cai, Georgios Tzimiropoulos, and Maja Pantic. End-to-end audiovisual speech recognition. *ICASSP*, 2018.

[125] Jesús Andrés Portillo-Quintero, José Carlos Ortiz-Bayliss, and Hugo Terashima-Marín. A straightforward framework for video retrieval using clip. In *MCPR*, 2021.

[126] Ariana Tulus Purnomo, Ding-Bing Lin, Tjahjo Adiprabowo, and Willy Fitra Hendria. Non-contact monitoring and classification of breathing pattern for the supervision of people infected by covid-19. *Sensors (Basel, Switzerland)*, 21, 2021.

[127] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. IEEE*, 77:257–286, 1989.

[128] Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[129] Alexander Richard, Hilde Kuehne, and Juergen Gall. Weakly supervised action learning with rnn based fine-to-coarse modeling. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1273–1282, 2017.

[130] Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. A dataset for movie description. In *CVPR*, 2015.

[131] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65 6:386–408, 1958.

[132] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[133] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

[134] Ramon Sanabria, Ozan Caglayan, Shruti Palaskar, Desmond Elliott, Loïc Barrault, Lucia Specia, and Florian Metze. How2: a large-scale dataset for multimodal language understanding. In *Proceedings of the Workshop on Visually Grounded Interaction and Language (ViGIL)*. NeurIPS, 2018.

[135] J. Saunders. Real-time discrimination of broadcast speech/music. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 2, pages 993–996 vol. 2, 1996.

[136] Eric D. Scheirer and Malcolm Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2:1331–1334 vol.2, 1997.

[137] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. In *INTERSPEECH*, 2019.

[138] Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152, 2012.

[139] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *ArXiv*, abs/1508.07909, 2016.

[140] Paul Hongsuck Seo, Arsha Nagrani, Anurag Arnab, and Cordelia Schmid. End-to-end generative pretraining for multimodal video captioning. *CVPR*, 2022.

[141] Paul Hongsuck Seo, Arsha Nagrani, and Cordelia Schmid. Look before you speak: Visually contextualized utterances. In *CVPR*, 2021.

[142] Dmitriy Serdyuk, Otavio Braga, and Olivier Siohan. Audio-visual speech recognition is worth 32x32x8 voxels. *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2021.

[143] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, 2014.

[144] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

[145] Tejas Srinivasan, Ramon Sanabria, and Florian Metze. Analyzing utility of visual context in multimodal speech recognition under noisy conditions. *The How2 Challenge: New Tasks for Vision & Language, ICML*, 2019.

[146] Tejas Srinivasan, Ramon Sanabria, and Florian Metze. Looking enhances listening: Recovering missing speech using images. *ICASSP*, 2020.

[147] Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Learning video representations using contrastive bidirectional transformer. *arXiv 1906.05743*, 2019.

[148] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. VideoBERT: A joint model for video and language representation learning. In *ICCV*, 2019.

[149] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *NeurIPS*, 2014.

[150] Gabriel Synnaeve, Qiantong Xu, Jacob Kahn, Edouard Grave, Tatiana Likhomanenko, Vineel Pratap, Anuroop Sriram, Vitaliy Liptchinsky, and Ronan Collobert. End-to-end asr: from supervised to semi-supervised learning with modern architectures. *ArXiv*, abs/1911.08460, 2019.

[151] Satoshi Tamura, Hiroshi Ninomiya, Norihide Kitaoka, Shin Osuga, Yurie Iribe, K. Takeda, and Satoru Hayamizu. Audio-visual speech recognition using deep bottleneck features and high-performance lipreading. *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2015.

[152] Silero Team. Silero vad: pre-trained enterprise-grade voice activity detector (vad), number detector and language classifier. https://github.com/snakers4/silero-vad, 2021.

[153] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[154] Michael Wray, Diane Larlus, Gabriela Csurka, and Dima Damen. Fine-grained action retrieval through multiple parts-of-speech embeddings. In *ICCV*, 2019.

[155] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv 1609.08144*, 2016.

[156] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *ECCV*, 2018.

[157] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell. Distance metric learning, with application to clustering with side-information. In *NeurIPS*, 2002.

[158] Hu Xu, Gargi Ghosh, Po-Yao Huang, Dmytro Okhonko, Armen Aghajanyan, Florian Metze, Luke Zettlemoyer, and Christoph Feichtenhofer. VideoCLIP: Contrastive pre-training for zero-shot video-text understanding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

[159] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. MSR-VTT: A large video description dataset for bridging video and language. In *CVPR*, 2016.

[160] Qiantong Xu, Alexei Baevski, Tatiana Likhomanenko, Paden Tomasello, Alexis Conneau, Ronan Collobert, Gabriel Synnaeve, and Michael Auli. Self-training and pre-training are complementary for speech recognition. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3030–3034, 2021.

[161] Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Just ask: Learning to answer questions from millions of narrated videos. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1666–1677, 2021.

[162] Youngjae Yu, Jongseok Kim, and Gunhee Kim. A joint sequence fusion model for video question answering and retrieval. In *ECCV*, 2018.

[163] Youngjae Yu, Hyungjin Ko, Jongwook Choi, and Gunhee Kim. End-to-end concept word detection for video captioning, retrieval, and question answering. *CVPR*, 2017.

[164] Neil Zeghidour, Olivier Teboul, Félix de Chaumont Quitry, and Marco Tagliasacchi. Leaf: A learnable frontend for audio classification. *ICLR*, 2021.

[165] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014.

[166] Bowen Zhang, Hexiang Hu, and Fei Sha. Cross-modal and hierarchical modeling of video and text. In *ECCV*, 2018.

[167] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. In *ECCV*, 2016.

[168] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1:43–52, 2010.

[169] Yu Zhang, James Qin, Daniel S. Park, Wei Han, Chung-Cheng Chiu, Ruoming Pang, Quoc V. Le, and Yonghui Wu. Pushing the limits of semi-supervised learning for automatic speech recognition. *ArXiv*, abs/2010.10504, 2020.

[170] Bolei Zhou, Àgata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Trans. PAMI*, 40:1452–1464, 2018.

[171] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *AAAI*, 2018.

[172] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *CVPR*, 2018.