Spatio-Temporal Facial Expression Recognition with 3D Convolutional Neural Networks

by

Théo AYRAL

THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT OF A MASTER'S DEGREE
WITH THESIS IN AUTOMATED MANUFACTURING ENGINEERING
M.A.Sc.

MONTREAL, JUNE 10, 2021

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Éric Granger, Thesis supervisor
Department of Systems Engineering, École de technologie supérieure

Mr. Marco Pedersoli, Co-supervisor
Department of Systems Engineering, École de technologie supérieure

Mr. Simon Bacon, Co-supervisor
Department of Health, Kinesiology, and Applied Physiology, Concordia University

Mr. Mohamad Forouzanfar, President of the board of examiners
Department of Systems Engineering, École de technologie supérieure

Mrs. Samira Ebrahimi-Kahou, Member of the jury
Department of Software and Information Technology Engineering, École de technologie supérieure

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON MAY 10, 2021

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

I thank the Canadian Institutes of Health Research (SMC-151518) for their financial support and especially Dr. Simon Bacon from Concordia University, as our discussions provided grounding for this work with real-world scenarios. Thanks also to Compute Canada, Calcul Québec, for providing the computational resources which made experimentation possible.

For their supervision, I thank Éric Granger, always pushing me further in discovering new literature and following strict methodology, and Marco Pedersoli for bringing theoretical foundations and experimental ideas. I am also grateful for the advice and enthusiasm of other members of LIVIA.

I thank my friend Marc for having played a driving role in the development of my interest in deep learning. I thank my family for their support in moving to Montreal. I thank Maria for providing me with an inspiring and active environment, and my friends for their presence in these times of sanitary crisis and social distancing.

Réseaux de Neurones Convolutifs 3D pour la Reconnaissance Spatio-Temporelle d'Expressions Faciales

Théo AYRAL

RÉSUMÉ

Notre étude concerne les réseaux de neurones convolutifs 3D (3D Convolutional Neural Networks, 3D-CNNs) pour la reconnaissance d'expressions faciales (Facial Expression Recognition, FER) dans les vidéos. Au cours des dernières années, l'apprentissage profond s'est imposé comme un principe majeur pour le développement de systèmes automatiques pour la FER. La transition de l'effort de recherche vers les systèmes d'apprentissage profond s'accompagne aussi d'une régression vers les méthodes spatiales, les modèles étant de plus en plus dépendants de la quantité de données d'entraînement, ce qui favorise les bases de données d'images 2D par rapport aux vidéos qui sont plus difficiles à collecter, annoter et analyser.

Dans ce mémoire, différentes approches pour la FER spatio-temporelle sont évaluées, utilisant des modèles d'apprentissage profond pré-entraînés. L'étude se concentre notamment sur le principe de convolutions 3D pour la classification de vidéos, questionnant la pertinence d'un traitement unifié des dimensions spatiales et temporelle, considérant les vidéos comme des volumes de données 3D. Pour limiter le coût de calculs et la consommation de mémoire, et pour gérer la relative rareté des données annotées, l'entraînement des réseaux 3D est généralement réalisé avec des clips vidéos très courts, extraits depuis les vidéos d'entraînement. Une nouvelle méthode est proposée pour l'amélioration des performances des modèles 3D. Le softmax stochastique temporel ainsi développé est basé sur une pondération temporelle du mécanisme de sélection des clips d'entraînement. Cette méthode permet au modèle de se concentrer sur les clips les plus pertinents, résultant dans l'amélioration de l'efficacité de l'entraînement et des performances de classification. Des expériences sont réalisées sur différentes tâches de classification vidéo, principalement en FER, montrant la pertinence d'une telle pondération des mécanismes d'échantillonnage et d'agrégation temporels pour répondre aux problèmes habituels tels que l'occlusion, le découpage imprécis et l'annotation grossière des vidéos, et la distribution inégale de l'information pertinente dans les vidéos au cours du temps. De plus, l'étude se porte sur les mécanismes d'attention visuelle, pour opérer des pondérations plus complexes inhibant ou renforçant certaines régions de vidéo. Les différents types d'attention qui peuvent être développés pour les 3D-CNNs sont analysés, clarifiant leur pertinence pour la reconnaissance spatio-temporelle et la FER en particulier. Les expériences montrent notamment l'intérêt d'une représentation contextuelle, intégrant le contenu global de la vidéo, pour guider l'attention sur certaines positions. À travers ces thèmes, l'étude porte sur l'importance relative des frames temporelles dans une vidéo, pour répondre à la distribution inégale de l'information pertinente dans le temps.

Mots-clés: informatique affective, reconnaissance d'expressions faciales, reconnaissance spatio-temporelle, apprentissage profond, 3D CNNs, attention visuelle

Spatio-Temporal Facial Expression Recognition with 3D Convolutional Neural Networks

Théo AYRAL

ABSTRACT

In this thesis, focus is set on spatiotemporal 3D convolutional neural networks (3D CNNs) for facial expression recognition (FER) in videos. Over the last decade, deep learning has emerged as a state-of-the-art paradigm for FER and spatiotemporal recognition. The transition of research focus toward deep learning was also a regression from spatiotemporal to spatial methods. Models are increasingly dependent on the quantity of training data, favouring 2D-image datasets over videos which are more difficult to collect, label and process.

Different approaches to spatiotemporal FER are evaluated, leveraging pretrained deep-learning models. The 3D-convolution paradigm for video classification is analyzed, questioning the relevance of considering spatial and temporal dimensions of video data as forming a unified 3D volume. To cope with the computational requirements and scarcity of data, clip sampling is commonly adopted for training 3D CNNs. To increase performance within this framework, a new method is developed. The proposed temporal stochastic softmax is based on a weighted clip-sampling mechanism. This method allows the model to focus on the most relevant clips, for efficient training and accurate recognition. Experiments are carried out on several video classification tasks, focusing on facial expression recognition, and discussions are provided concerning the relevance of such weighted temporal sampling and pooling mechanisms in addressing common issues such as occlusion, inaccurate trimming and coarse annotation of videos, and uneven distribution of discriminant cues across time. In addition, the study explores visual attention mechanisms, as a way to implement more complex weighting behaviors for masking or highlighting regions of the input videos. The different attentional behaviors that can be developed with 3D CNNs are analyzed, and their relevance is discussed in the context of spatiotemporal recognition and FER specifically. Experiments notably demonstrate the benefits of guiding attention with contextual representations, summarizing the global information of the video. The study discusses the relative importance of temporal frames in a video, to address the heterogeneous distribution of relevant cues in time. The proposed methods increase the performance of 3D CNNs on all benchmarks, providing better ways to learn from data. Such methods for efficient spatiotemporal recognition should become more and more important as larger datasets become available in the future, allowing richer training of 3D CNNs.

Keywords: affective computing, facial expression recognition, spatiotemporal recognition, deep learning, 3D CNNs, clip sampling, visual attention

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Page

LIST OF ABBREVIATIONS

AU	Action Unit
AV	Audio-Video
CNN	Convolutional Neural Network
DL	Deep Learning
DRAW	Deep Recurrent Attention Writer
EER	Equal Error Rate
FER	Facial Expression Recognition
FLDA	Fisher's Linear Discriminant Analysis
fps	frames per second
GAP	Global Average Pooling
GPU	Graphics Processing Unit
HMM	Hidden Markov Model
LBP	Local Binary Patterns
LSTM	Long Short-Term Memory
MIL	Multiple Instance Learning
MLP	Multi-Layer Perceptron
NMT	Neural Machine Translation
OPI	Observed Pain Intensity
PCA	Principal Component Analysis

RAM	Recurrent Attention Model
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
ROI	Region Of Interest
SGD	Stochastic Gradient Descent
SIFT	Scale-Invariant Feature Transform
STN	Spatial Transformer Network
SVM	Support Vector Machine

INTRODUCTION

As images before them, videos are receiving a lot of attention from the deep learning research community. New methods of video analysis, classification or captioning are very promising and already have applications in robotics, surveillance, marketing and health monitoring. The performance of Convolutional Neural Networks (CNNs) on image classification and related computer vision problems made them a very natural choice to tackle video classification. The naive approach consists in analysing videos as a set of independent frames. These methods, based exclusively on spatial information, are not yet outperformed by spatiotemporal methods on all challenging domains. This reveals a failure in taking advantage of the dynamics component. Our study aims at understanding how spatiotemporal information can be conjointly leveraged to classify emotion videos.

This thesis is focused on the problem of facial expression recognition (FER) from videos, and more specifically on emotion classification. Facial expressions are a key component of non-verbal communication, allowing humans to share their emotions and intentions. Automatic FER has a lot of applications in marketing, education, security and behavioral medicine among other fields.

Facial expression recognition aims at inferring a person's mental state from visual cues. In this sense, we consider emotion recognition and pain detection as specific tasks involving FER. As this thesis is focused on computer vision methods and facial expression recognition, the notions of emotion recognition and pain detection only refer to the FER aspect of these tasks. We do not consider other data modalities such as text, sound or physiological signals. Working with facial expressions is one of the most promising approaches to automatic affect recognition, as it requires relatively simple data acquisition. FER can be leveraged with a simple capture device, and thus can be used for real-world large-scale applications, because of the availability of webcams, for mobile applications with smartphone, video conference tools, in

the public areas with security cameras, in home situations, etc. Providing an effective way to automatically detect and objectively measure emotions, depression or pain, can have very varied applications, with possibilities in industry to evaluate marketing strategies, product development and user experience. Also in the education domain, FER can help design learning platforms and educational tools, by analyzing student behavior. More importantly, there are numerous possible evolutions for medicine applications. FER can be leveraged to design online assessment and training tools for behavior-change counselling, by and for physicians. Witnessing the emergence of automatic systems for recognition of facial expression, Pantic & Rothkrantz (2000b) note that "such a system could also make classification of facial expressions widely accessible as a tool for research in behavioral science and medicine". As stated by Chen, Ansari & Wilkie (2018b), "Facial pain expression is an important modality for assessing pain, especially when the patient's verbal ability to communicate is impaired". Another key application which constitutes an important research domain is related to social robots, and human-machine interactions.

It is however a difficult problem because of the complexity of creating databases for training and validation of models. Emotions can't be induced, controlled or captured easily to generate training data for deep learning models. Also, defining emotions objectively and identifying expressions without contextual information make the annotation process problematic. Generally, spatiotemporal information has proven to be difficult to leverage in emotion videos because of the lack of labelled data, the poor signal-to-noise ratio in the videos, high intra-class variations (e.g. different persons expressing the same emotion) and low inter-class variations (e.g. the same person expressing different emotions).

Human communication is multimodal, and facial expressions are only one aspect used to visually communicate emotions, pain or other mental states. Emotions or related mental states can be identified through several modalities of data, including text (Dyer & Kolic, 2020), sound (Ringeval, 2011), image (Vielzeuf, Kervadec, Pateux, Lechervy & Jurie, 2018), spatial

3D data (Li, Huang, Li & Wang, 2018b), infrared images He, Wang, Lan, Fu & Ji (2013), physiological signals (Bulagang, Weng, Mountstephens & Teo, 2020), describing several types of phenomenons, as facial expressions, voice, heartbeat, skin conductance or posts on social media. As discussed in the work of Pantic & Rothkrantz (2000b), facial expressions are a major vector of communication, and play a key role in the interpretation of a vocal message, and in coordinating a conversation. "Personality, attractiveness, age, and gender can also be seen from someone's face", Pantic & Rothkrantz (2000b).

Humans detect and recognize faces, and interpret facial expressions with little effort, as a key component of their interactions. The human visual system, and its ability to recognize facial expressions is often used to guide research for developing new automated methods and set expectations for their performance. "It is generally believed that two-gray-levels images of 100 to 200 pixels form a lower limit for detection of a face by a human observer", Pantic & Rothkrantz (2000b). On the other hand, interpretation of these expressions is a more complex task, as expressions are subjective and their interpretation can dependent on context information (Bassili, 1978).

Research effort has been made to determine the most relevant cues to recognize and interpret facial expressions, and to detect, encode and classify them. This question spans domains of biology (Darwin, 1998), psychology (Ekman & Friesen, 1971), neuroscience (Freiwald, Tsao & Livingstone, 2009), computer science (Sariyanidi, Gunes & Cavallaro, 2015) and robotics (Yu & Tapus, 2019). Some research domains focus on studying the human vision system or social interactions, while computer science tries to imitate it, or develop other models to encode emotions. Robotic systems even produce facial expressions for human-robot interactions.

Various formulations of the emotion recognition problem are used in literature, with different ways of encoding facial expressions:

– the Facial Action Coding System (FACS) designed by Ekman, Friesen & Hager (2002), describes expressions with the presence of action units,
– classification among six universal emotions (happiness, sadness, fear, anger, surprise and disgust), as identified by Ekman & Friesen (1971) and Darwin (1998),
– a model of affect in continuous dimensions, as valence, arousal, power and expectation (Gunes & Schuller, 2013; Russell, 1980).

As explained by Valstar & Pantic (2012), this corresponds to different approaches to facial expression measurement in psychological research (Ekman & Friesen, 1969):

– the message-level approach considers facial-affect detection with the objective of inferring the high-level psychological state related to a displayed facial expression;
– the sign-level approach encodes the low-level, elementary features of expressions, usually in the framework of FACS.

Deep learning can be considered in between these categories, building hierarchies of features through end-to-end training with emotion labels. Learned features can be seen as a data-driven alternative to the standard action units.

The framework of FACS, proposed by Ekman *et al.* (2002), is widely accepted to describe facial expressions. Expressions are understood as combinations of facial action units (AUs), with contractions of muscles modifying the shape and location of facial components. Actions are describe by their type, intensity and timing, with anatomical considerations, and independently of their meaning and interpretation (Cohn & Ekman, 2002). This system provides a standard protocol for researchers to describe facial expressions objectively. FACS coding by human experts is a very time-consuming task that usually prohibits its clinical use, and automated coding systems are a focus of research.

The categorization of facial expressions in one of six universal emotion categories is considered the most popular approach in the survey of Li & Deng (2018). However, elicited emotions are generally a blend of these basic emotions, or even out of the scope of these six basic emotions. "Pure" emotion categories are not relevant to real-world scenarios (Pantic & Rothkrantz, 2000b). A more complex model based on compound expressions is proposed by Du, Tao & Martínez (2014). Also, the cross-culture, universal aspect of these expressions has been questioned (Jack, Garrod, Yu, Caldara & Schyns, 2012).

Inconsistencies in problem definitions across datasets are highlighted in the work of Li & Deng (2018). Each dataset is designed with a certain goal and a particular vision of the task, and the annotation is tailored according to it. For instance some datasets use seven basic emotions when others use six. This makes it difficult to leverage several datasets to enlarge the training data. When this is done, it usually consists in a multi-task setting (Kaiser, Gomez, Shazeer, Vaswani, Parmar, Jones & Uszkoreit, 2017; Zeng, Shan & Chen, 2018).

There are also different ways to model facial expressions in time. “The temporal evolution of an expression is typically modelled with four temporal segments [...]: neutral, onset, apex and offset. Neutral is the expressionless phase with no signs of muscular activity. Onset denotes the period during which muscular contraction begins and increases in intensity. Apex is a plateau where the intensity usually reaches a stable level; whereas offset is the phase of muscular action relaxation” (Sariyanidi *et al.*, 2015). More simplistic approaches only describe the presence of an expression in a temporal segment, without explicitly modelling intensities.

Datasets regroup images, videos or other type of data samples, in a coherent way to train and validate automatic FER systems. Each data sample is annotated with a target label, that is the expected response of the automatic system when processing the given sample. Thus, through their annotation system, datasets are usually restricted to certain tasks. For instance, if a dataset provides emotion-class labels, it will be possible to train a model for emotion classification,

but not to detect specific AUs in the images. As such, datasets are typically designed for a specific task, and with a clear formulation of the FER problem. This is also true for open research competitions, which define a challenge with a standardized evaluation method in order to compare submissions, based on a specific dataset. For instance, the AVEC challenge (Valstar, Schuller, Smith, Eyben, Jiang, Bilakhia, Schnieder, Cowie & Pantic, 2013), consists in continuous affect recognition, with valence and arousal dimensions. Alternatively, the FERA AU challenge (Valstar, Jiang, Mehu, Pantic & Scherer, 2011) requires the detection of FACS action units in videos. These can be considered as different formulations of the same task, i.e. facial expression recognition. An example of related but different tasks is the classification of real and fake expressions, which also require specific datasets (Wan, Escalera, Anbarjafari, Escalante, Baró, Guyon, Madadi, Allik, Gorbova, Lin & Xie, 2017).

Encoding and recognition of expressions is also difficult because of their subjective aspect. As discussed in Pantic & Rothkrantz (2000b), " each person has his/her own maximal intensity of displaying a particular facial expression", and "the interpretation of the body language is situation-dependent". To illustrate this complexity, we can observe the difference in designs of two popular pain detection datasets. The UNBC-McMaster Shoulder Pain Expression Archive Database (Lucey, Cohn, Prkachin, Solomon & Matthews, 2011) tries to objectively describe the expression with a level of observed pain intensity. BioVid Heat Pain Database (Werner, Al-Hamadi, Niese, Walter, Gruss & Traue, 2013) proposes to categorize the intensity-level of the heat stimulus that induced the pain expression. BioVid also provides stimulus levels tuned to each participant's pain-acceptance threshold.

In controlled scenarios, data is usually captured in a lab, with frontal view of the face. Expressions can be posed or elicited. This is typically opposed to "in-the-wild" capture, presenting real-world data, and natural expressions. The survey Sariyanidi *et al.* (2015) reports that in 2015: "Most affect recognisers are validated on posed datasets, which differ from naturalistic datasets in

terms of illumination conditions, head-pose variations and nature of expressions (subtle vs. exaggerated [...])". Although, these categories are not clearly defined. For instance, the EmotiW affective computing challenge focuses on "in-the-wild" scenarios (Dhall, Kaur, Goecke & Gedeon, 2018). However, the Emotion-Recognition subchallenge is based on the dataset of Acted Facial Expressions in the Wild (AFEW, presented in Dhall, Goecke, Lucey & Gedeon, 2012), which regroups emotional video samples extracted from movies. These facial expressions are professionally posed, and with varying scenes, backgrounds, body movements and lighting conditions, which are expected to provide close to real-world data. In contrast, the subchallenge of Student-Engagement Prediction is based on a dataset of videos collected with actual students watching online courses in various conditions and places (Kaur, Mustafa, Mehta & Dhall, 2018).

In the domain of pain detection from facial expressions, we can again compare two popular datasets. The UNBC-McMaster Shoulder Pain Expression Archive Database (Lucey *et al.*, 2011) provides detailed annotations for each frame, manually coded AUs and pain intensity score. However, it only contains 200 videos. On the other hand, BioVid Heat Pain Database (Werner *et al.*, 2013) contains 8700 videos, but only provides sequence-level labels of pain-stimulus intensity. Also, BioVid was collected with videos of only 87 participants, with 100 videos per subject, resulting in a lack of diversity in the samples. This shows the trade-off in quantity of data and quality of annotation.

Another issue of datasets is the class-imbalance, which will bias the results toward the most represented classes, depending on the dataset construction (with some emotions being more difficult to elicit or capture than others). Data augmentation and weighting of the loss function can help address this issue.

Surveying literature for 3D spatial (with depth) FER in Alexandre, Soares & Thé (2020), the authors note that even though "learned features consistently demonstrate superior recognition rates compared to handcrafted ones", datasets of 3D face scans are very limited to train

deep learning models. Deep approaches have to leverage pretrained deep models and data augmentation to cope with the limited training data. This issue does not only concerns the niche domain of 3D spatial FER. Most surveys related to automatic FER note the lack of training and validation data as a limiting factor (Sariyanidi *et al.*, 2015; Chen *et al.*, 2018b; Li & Deng, 2018), especially for the development of deep learning models. As discussed by Li & Deng (2018), approaches to build large-scale, annotated datasets for deep learning include crowd-sourcing and automatic labeling tools. Notably, EmotioNet (Benitez-Quiroz, Srinivasan & Martínez, 2016), and AffectNet (Mollahosseini, Hassani & Mahoor, 2019), respectively regroup a million and 450,000 images automatically collected from the World Wide Web and annotated automatically or manually with crowd-sourcing. The AFEW video dataset was also collected and labelled semi-automatically (Dhall *et al.*, 2012), and the number of video samples is in the order of a thousand. Another research direction is training from synthetic data. In the context of FER, this is studied in Kortylewski, Schneider, Gerig, Egger, Morel-Forster & Vetter (2018); Abbasnejad, Sridharan, Tien, Denman, Fookes & Lucey (2017).

Automatic FER systems have to deal with uneven scales and orientations of faces in images. A typical challenge with visual data is occlusion, which can come from the subject's hair and glasses or from other elements in the scene. Another recurrent issue in literature is the confusion between expression-related features and those linked to the subject's identity (Sariyanidi *et al.*, 2015). Li & Deng (2018) explain that "high inter-subject variations exist due to different personal attributes, such as age, gender, ethnic backgrounds and level of expressiveness". Facial expression recognition from images constitutes a classification problem in which data presents high intra-class variations (e.g. different subjects having different face features for the same expression class) and high inter-class similarity (e.g. the same face with different expressions). Proposed solutions include studying the difference between an input image and the subject's neutral pose. Also, spatiotemporal features, with the motion of face features, are considered to be shared across individuals, and more robust to identity bias.

This study focuses on 3D convolutional neural networks (3D CNNs) as they have been shown to be powerful models for spatiotemporal recognition and classification, and yet a lot remains to be clarified about efficient implementations and their capabilities both theoretically and in practice. Especially, the use of 3D CNNs for facial expression recognition is not well studied, as opposed to action recognition. We study how to build better spatiotemporal representations of videos with unified 3D models (Chapter 2), by improving the training process (Chapter 3) and by enhancing the architecture with additional modules implementing spatiotemporal visual attention (Chapter 4).

Our study revolves around the idea of unified spatiotemporal analysis, that is jointly processing spatial and temporal dimensions. However, space and time cannot always be treated equally. Throughout our study, we discuss the differences in the processing of spatial and temporal dimensions, and the reasons and motivations, whether they are inherent to the task or simply imposed by implementation or resources.

To present the background of our study, Chapter 1 provides a discussion of related work and the current state of the art for spatiotemporal deep learning and facial expression recognition. This chapter describes challenges in processing videos for classification, and the theoretical foundations we work with. We observe that our problem is not solved yet and define the objectives of our study.

In Chapter 2, we compare different spatiotemporal architectures to perform emotion classification from videos. There are mainly two types of architectures for spatiotemporal deep learning models: spatial 2D CNN coupled with a Recurrent Neural Network (RNN), or spatiotemporal 3D CNN. For both the 2D and the 3D approaches, using models pre-trained on larger datasets has proven necessary. The effectiveness of RNNs has been widely studied, but they only model patterns in the evolution of high-level spatial features extracted by CNNs. Alternatively, 3D convolutions exploit the relationships between spatiotemporal features, finding patterns

directly in the video dynamics. They consider data volumes composed of three dimensions encoding appearance and motion information. With the apparition of larger video datasets, 3D convolutional architectures have already achieved impressive results and started dominating the state of the art on action recognition challenges. Considering their increasing performance, still limited by the lack of knowledge about the architectures and methods to cope with poor training data, and motivated by the theoretical suitability of the 3D spatiotemporal approach for facial expression video analysis, we evaluate different modalities of 3D convolutional architectures.

Chapter 3 presents a new training method for 3D CNNs, based on temporal softmax weighting of video clips. Spatiotemporal models come with interesting challenges due to their size, with computational requirements necessitating specific training methods. For practical reasons, 3D Convolutional Neural Networks (3D CNNs) are usually trained with relatively short clips randomly extracted from videos. However, such uniform sampling is generally sub-optimal because equal importance is assigned to each temporal clip. Thus, we develop a temporal stochastic softmax strategy to efficiently train 3D models for facial expression recognition with short video clips. It relies on softmax temporal pooling and a weighted sampling mechanism to select the most relevant training clips. Experimental results obtained with the proposed method on several facial expression recognition benchmarks show the benefits of focusing on more informative clips in training videos. The method improves classification performance, by providing richer training data, and highlighting discriminant information during testing.

Chapter 4 discusses attention mechanisms as a way to further develop the model's ability to focus on regions of interest. The studied models determine the relative importance of all locations in the video, spatially and temporally. Attention is an active domain of deep-learning research although its definition is not clear. Lots of strategies have been proposed to model spatial and temporal attention as a component of a deep learning architecture. However, modelling spatiotemporal attention directly is still an open question. We experiment and discuss this issue.

CHAPTER 1

RELATED WORK

1.1 Automatic facial expression recognition

1.1.1 FER pipeline

Figure 1.1 represents the traditional automatic FER pipeline, with four main components: preprocessing, face detection, feature extraction and classification. The formulation of this pipeline can vary, more detailed descriptions are proposed in surveys such as Sariyanidi *et al.* (2015) or Li & Deng (2018).

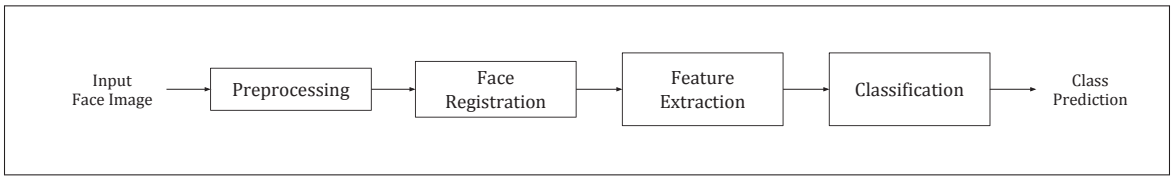

Figure 1.1 Traditional pipeline for automatic FER systems

Preprocessing aims at facilitating face detection and the feature extraction process by reducing illumination variations for instance. On-the-fly data augmentation is also a typical preprocessing step for deep-learning approaches (Li & Deng, 2018). Face detection allows to crop the image around the interest region, improving signal-to-background ratio. Face alignment can also be used to reduce head-pose variations using facial landmarks. Face detection and alignement can also be considered as preprocessing steps.

The feature extraction process aims at describing the image with discriminative features in a low-dimensionality representation, facilitating the classification with higher class-separability (low intra-class variation, high inter-class variations). Traditional computer vision methods rely on hand-crafted features (as opposed to learned features) and integrate expert domain knowledge. On the other hand, the deep learning approach uses data-driven optimization to learn convolution kernels that can recognize spatiotemporal patterns in the video. Stacks of

convolutional layers extract a hierarchy of features from the input, to build a new representation. These representations are then categorized into one of the possible expression classes.

1.1.2 Face detection

Facial expression recognition applications can work with facial images or arbitrary images (Pantic & Rothkrantz, 2000b). In both cases, recognition systems require to identify the position or contour of the face, or even more precise localization of facial features, as the position of irises or eye outer corners, the height of mouth, etc.

Whole Face Registration, parts registrations, or points registration to classify shapes directly or to transform the face image to canonical pose. Input images are usually aligned to a prototypical face with landmark-based approaches. Detecting fiducial points allows to isolate regions of interest (ROIs) that will be relevant to classification (removing background but also non-deformable parts of the face). It is also possible to locally warp parts of the landmarks to modify the facial expression, for instance in the aim of obtaining the neutral expression of the subject. Also, isolating ROIs in the face image allows to extract features only in these relevant parts, usually the mouth and eyes, contrary to global approaches which extract features by processing the entire face area.

Most systems perform spatial face registration on a stack of frames. This can cause issues in the temporal consistency of face registration. With consequences in representations obtained with low-level spatiotemporal feature-extraction. Alternatively, faces can be temporally tracked in the scene, from an initial spacial localization.

Sariyanidi *et al.* (2015) note that facial registration is usually employed to remove head-pose variation, but some approach consider head pose as a relevant feature of affective behavior that needs to be modelled.

From the survey of Pantic & Rothkrantz (2000b), we can see that, in the late 1990s, the most popular face detection methods involve canny edge detectors, point distribution models,

brightness distributions, color detectors, convex region detectors and spatiotemporal filtering methods.

In 2015, Sariyanidi *et al.* (2015) observes the use of other methods: active appearance model (AAM), SIFT-flow, Robust FFT, and Lucas-Kanade approaches. Another popular detection framework is the Viola–Jones algorithm Viola & Jones (2001) based on the Integral Image representation and the AdaBoost learning algorithm with a cascade architecture, which can perform face recognition in real time.

In addition, the survey of Li & Deng (2018), shows that state-of-the-art face-alignement methods now include models entirely based on deep-learning. Deep models can perform face detection (Yang, Luo, Loy & Tang, 2015), face alignment with coarse-to-fine auto-encoder networks (CFAN presented in Zhang, Shan, Kan & Chen, 2014), or both tasks jointly as the multi-task CNN (MTCNN proposed in Zhang, Zhang, Li & Qiao, 2016a). Frontal views of input face images can also be synthesized by generative adversarial networks (GAN), for pose-invariant FER (Lai & Lai, 2018).

1.1.3 Feature representation

Pantic & Rothkrantz (2000b) compare analytic and holistic face representations. Analytic represenations consider the face as a set of features. The human visual system is considered to be closer to a holistic model, perceiving the face as a whole unit, considering the geometrical relationship of features.

Feature extraction approaches can be based on appearance, with textural information, considering pixel-value intensities, or based on shapes, considering the geometry of the facial features. In 2015, Sariyanidi *et al.* (2015) report: “The most notable recent trend is moving from shape to appearance representations”. Also, “a practice that proved particularly useful is using shape representations in conjunction with appearance representations, combining various types of configural, holistic and componential information”. They explain that this corresponds to the human vision system.

In geometry-based analytic face representations, the face is modeled as a set of facial points, whose movement and relationships can be analyzed. Geometry-based features can be obtained by simply concatenating the coordinates of facial landmarks, or encoding distances between these points. This type of design was motivated by Johansson's point-light display experiments (Johansson, 1973), demonstrating that gestures could be recognized with only a few moving points. Several research works focused on classifying facial expressions from manually extracted geometric features (e.g. facial landmarks).

Spatiotemporal geometric features from tracked facial points provide a facial expression representation very close to the AU system, allowing to directly leverage expert knowledge from cognitive science in the analysis of muscular activity, described by the temporal variations in position of fiducial points.

In the survey of Sariyanidi *et al.* (2015), popular low-level appearance-based feature-extraction methods include local binary patterns (LBP as used in Shan, Gong & McOwan, 2009), local phase quantization (LPQ) or scale-invariant feature transform (SIFT). Information can be encoded with histograms of these low-level features, or Gabor represenations with manually designed filters. LBP-TOP, i.e. LBP with three orthogonal planes, or volume LBP as in Zhao & Pietikäinen (2007), provides a spatiotemporal adaptation of this low-level feature extraction.

High-Level representations can be obtained, for example, with non-negative matrix factorization and sparse coding based on action-unit dictionaries (Zhi, Flierl, Ruan & Kleijn, 2011; Zhong, Liu, Yang, Huang & Metaxas, 2015).

Dimensionality-reduction techniques are also employed to reduce the size of the representation for lighter computation, but also provide invariant to illumination or answer other challenges as registration errors and identity bias (Sariyanidi *et al.*, 2015). This includes pooling, feature selection (boosting techniques, AdaBoost...) and feature extraction with principal component analysis (PCA).

As discussed in the work of Valstar & Pantic (2012), spatiotemporal analysis has been proven to be beneficial for tasks related to ours, such as recognition of posed and spontaneous expressions or categorization of complex psychological states like types of pain and mood. However, the relevance for standard expression recognition is not obvious. The experiments of Valstar & Pantic (2012) show that a low-level temporal analysis with a geometric representation (using characteristic facial points) can model changes in facial shape between frames, thus improving the detection of action units. This suggests that temporal analysis can play a role, not only on the high-level description of expression evolution through the video (to decide what emotion is predominant based on all the detected features) but also with the detection of local motion features (e.g. to detect a particular facial muscle movement).

Research has shown the importance of the temporal dimension for FER, be it through tracking of facial landmarks to analyse their displacements, or to directly detect local, low-level features (e.g. LBP-TOP as in Wang, Yu, Stevens & Liu, 2015). "Spatio-temporal representations consider a range of frames within a temporal window as a single entity, and enable modelling temporal variation in order to represent subtle expressions more efficiently. They can discriminate expressions that look similar in space", Sariyanidi *et al.* (2015).

This is why, even though some research tend to show that frame-level analysis can perform as well as current spatiotemporal models (as will be discussed in the next section), we consider that FER can benefit from leveraging motion features because of the inherent temporal structure of facial expressions, similarly to action recognition (Carreira & Zisserman, 2017). However, the relative failure of these deep spatiotemporal methods for FER, given their computational cost, shows that the domain is particularly difficult.

Also, videos can address issues of identity bias in FER. With a static image, the degree of facial muscle activations can be difficult to untangle from the subject's individual face characteristics. Leveraging video data can provide a neutral-expression image, which would help untangle expression features from identity features. Also, spatiotemporal features (e.g. representing the motion aspect of AUs), might be more robust to identity bias than purely spatial, appearance

features. We note that spatiotemporal expression features still highly depend on subject identity, in term of how subjects feel and express emotions. In addition to the identity differences, FER systems meet the problems of variation in illumination, pose, and skin tone, that are difficult to address with limited data.

1.1.4 Classification

In the survey of Pantic & Rothkrantz (2000b), classifiers are typically template-based, neural-network-based or rule-based. The idea is to compute distances in the feature-representation space, between the input image and a set of representative training facial images, or cluster representations. Spatiotemporal templates are obtained by averaging training images of a particular expression. Then a distance metric with the input image, in the feature space, is computed for classification. Only a few images are used for the templates (less than five).

Pantic & Rothkrantz (2000a) classify AUs, by computing the deformation of the model features between images with and without expression, for the same person. They use a hand-crafted, expert knowledge, rule-based method to then classify this AU-coded description into basic emotion categories.

Neural Networks can be used on top of several types of representations. They usually have one output neuron per category, whose value represent the probability of the input belonging to the corresponding expression category. Neural Networks usually use more training data, more than 100 images or image sequences.

In 2015 (Sariyanidi *et al.*, 2015), most affect recognition systems perform classification or recognition using machine learning techniques on the extracted features (Support Vector Machine, dynamic Bayesian network, conditional random fields, restricted Boltzmann machines, Hidden Markov Models).

The majority of systems surveyed in Alexandre *et al.* (2020), i.e. 3D-spatial FER studies published between 2013 and 2018, use of SVMs. This choice is usually justified by its high

performance in situations of limited data and high dimensional feature spaces. Other classifiers use Hidden Markov Models (HMMs) and Fisher's Linear Discriminant Analysis (FLDA). Less frequent methods include Random Forests, Nearest-neighbor and Artificial Neural Networks.

1.1.5 Deep learning models

This section provides an overview of possible methods based on deep neural networks for FER. Section 1.2 will be focused on surveying deep learning recognition systems in more depth.

1.1.5.1 Motivations

As explained by Li & Deng (2018), the "traditional" approach, based on handcrafted features and shallow classifiers, is limited when scaling from laboratory-controlled datasets to real world applications, with more variations of illumination and pose. On the contrary, deep learning approaches express their full potential on data captured "in the wild", with a lot of variations and examples. The issue becomes the availability of big labeled datasets for training.

The differential and localized nature of low-level features reduce the effect of illumination variations and registration errors. The abstraction power of high-level representations make them more efficient to tackle identity bias and produce "semantically interpretable" features Sariyanidi *et al.* (2015). Hierarchical representations combine both advantages, building encoding high-level information from low-level features. This can be achieved by using traditional high-level representations on top of hand-crafted low-level features extractors. However, deep learning architectures produce, by design, hierarchical representations, and are considered more efficient for this.

Li & Deng (2018) provide an insightful survey on the evolution of FER systems towards deep learning. This trend in FER follows the similar phenomenon on the tasks of image classification or action recognition (Krizhevsky, Sutskever & Hinton, 2012; Simonyan & Zisserman, 2015; Szegedy, Liu, Jia, Sermanet, Reed, Anguelov, Erhan, Vanhoucke & Rabinovich, 2015; Carreira & Zisserman, 2017; Hara, Kataoka & Satoh, 2018). With the creation of publicly

available relatively large datasets with labeled real-world data, research focus has transitioned "from lab-controlled to in-the-wild settings", and simultaneously from hand-crafted features to deep learning. The authors of Li & Deng (2018) also consider the role of "the dramatically increased chip processing abilities (e.g. GPU units) and well-designed network architecture", in establishing deep-learning models as state of the art.

1.1.5.2 Methods

The performance of deep learning for image classification is related to the development of convolutional neural network (CNN), as described by (Lecun, Bottou, Bengio & Haffner, 1998; Krizhevsky *et al.*, 2012). At each layer, feature extraction is performed with a set of learnable filters convolved across the entire image or feature map from the previous layer. With end-to-end training, filters are optimized in a data-driven approach to recognize relevant patterns. In comparision with multi-layer perceptrons (MLP), based on fully connected layers, the convolutional nature of CNNs corresponds to a parameter-sharing principle, which provides some degree of translation invariant, and is easier to optimize. CNNs consistute, by far, the most popular type of deep-learning models for FER, as can be measured in the survey of Li & Deng (2018).

Deep belief networks (Hinton, Osindero & Teh, 2006), originally designed as a stack of restricted Boltzmann machines (Hinton, Sejnowski et al., 1986), are very efficient to learn relevant features without supervision. Deep belief networks can detect regions of interest and capture variations in facial expressions. They have been proposed for generation and recognition of facial expressions, e.g. in Ranzato, Mnih, Susskind & Hinton (2013).

Instead of directly performing classification, deep autoencoders learn to reconstruct an input with minimal error (Hinton & Salakhutdinov, 2006). They perform dimensionality reduction, providing efficient data representations which can help classification or be leverage for generative tasks.

In order to model temporal dependencies in a sequence of frames, a very popular approach is using recurrent neural networks (RNNs). A well-established type of RNN is the long short-term memory (LSTM), introduced by Hochreiter & Schmidhuber (1997), with gating mechanisms controlling the input, recurrence and output flow of information. Alternatively, CNNs can be employed for temporal convolution, or satiotemporal convolution with three dimensional filters Ji, Xu, Yang & Yu (2010).

Generative Adversarial Networks (Goodfellow, Pouget-Abadie, Mirza, Xu, Warde-Farley, Ozair, Courville & Bengio, 2014) have rapidly demonstrated their efficiency for generation tasks. In the context of facial expressions, they are able to synthesize facial images with different expressions and poses Zhang, Zhang, Mao & Xu (2018).

Note that while CNNs can be considered as appearance-based feature extractors, geometric models are also possible with deep-learning approaches. In Jung, Lee, Yim, Park & Kim (2015), normalized *xy*-coordinates of facial landmarks are fed as input to a deep neural network for expression classification.

The study of Ciresan, Meier & Schmidhuber (2012) has shown that using multiple networks in parallel, and averaging their predictions allows the models to complement each others, resulting in improved performance. Such network ensembles, are very popular in FER, especially in challenge submissions. This allows for fusing models operating on different modalities of data, or simply to have multiple models for the same data, with different expertise and complexity. This is illustrated in the work of Vielzeuf, Pateux & Jurie (2017); Liu, Tang, Lv & Wang (2018a), with network ensembles comprising 2D CNNs working on individual frames, LSTM on frame-representation sequences, 3D CNN on video, fully connected neural network on audio data and SVM on facial landmarks. In terms of research objectives, it can be difficult to understand the individual contribution of each branch, and it makes any comparison between models hazardous.

Although an advantage of deep learning models is their end-to-end training scheme, with hierarchical representations up to the classifier, deep models can also be used as a part of a

recognition system, with other components based on traditional methods. Deep classifiers can be used on top of low-level feature extractors as in Luo, Chen, Takiguchi & Ariki (2017) or in Zhang, Zheng, Cui, Zong, Yan & Yan (2016b) with SIFT and deep neural networks. A very popular example is the use of optical-flow to model low-level motion features, as input to CNNs (Carreira & Zisserman, 2017). On the contrary, it has been proposed to leverage the hierarchical feature representations obtained with deep-learning models, with shallow classifiers. Tang (2013) explain that a linear support vector machine minimizes a margin-based loss instead of the cross-entropy usually used with CNNs.

Chen *et al.* (2018b) note that deep neural networks are very adapted to multi-label classification problems, which enabling an efficient modelling of AU combinations and co-occurences, as opposed to the "conventional" one-versus-all binary classifiers.

1.1.5.3 Challenges

With the transition to deep-learning recognition systems, FER literature generally agrees that the issues of expression-unrelated variations, such as illumination, head pose and identity bias still consitute an important challenge.

However, a new problematic has emerged, with the unprecedented requirements in training data. The most performant deep models, which are trained and validated on large action recognition datasets, quickly overfit to small facial-expression datasets. Li & Deng (2018) observe that “the existing facial expression databases are not sufficient to train the well-known neural network with deep architecture that achieved the most promising results in object recognition tasks”.

This supervision currently represents a major challenge. Data has to be measured in terms of quantity, diversity and quality of labelling. Some datasets are constituted from facial images taken from controlled environments, with participants posing specific expressions on demand Lyons, Akamatsu, Kamachi & Gyoba (1998). Such data can easily be labelled, as the ground-truth is known by construction, but the quality and realism of the expressions can be an issue. On

the other hand, annotation of "real-world" data, captured in the wild raise issues of cost and subjectivity, or even bias.

Different approaches have been proposed in the research literature to reduce the requirements on labels:

- semi-supervision and unsupervised approaches aim at leveraging more unlabelled data to produce efficient representations;
- weak-supervision provides partial labels, that are less precise than the desired output;
- it is also possible to directly focus on reducing the number of examples necessary, by improving the network's learning efficiency, by reducing the number of model parameters for instance.

With deep learning, it is often considered that issues related to subject specificity or illumination variation could be answered by providing more diverse training data. Models still require the mathematical tools and complexity to leverage such data.

Li & Deng (2018) explain that with the switch to deep learning approaches, there was a focus on spatial features, as recognition from static images is more convenient with deep models, due to training data requirements. Even when working with video data, frame-aggregation methods perform spatial feature extraction on each frame separately, with good results. Similar concerns are present for more general video recognition tasks (Sevilla-Lara, Zha, Yan, Goswami, Feiszli & Torresani, 2019), but are very limiting on FER.

This work is focused on the feature extraction, temporal aggregation and classification components. We study how deep spatiotemporal architectures can assume these three roles. Feature extraction and classification are implicit parts of a single component, a deep neural network trained end-to-end. A multi-stage architecture operates a hierarchical feature extraction, with the higher-level features corresponding to class scores. Also, we evaluate different ways to incorporate temporal analysis and aggregation within the deep-learning framework.

1.2 Deep learning for spatio-temporal recognition

1.2.1 Architectures for video classification

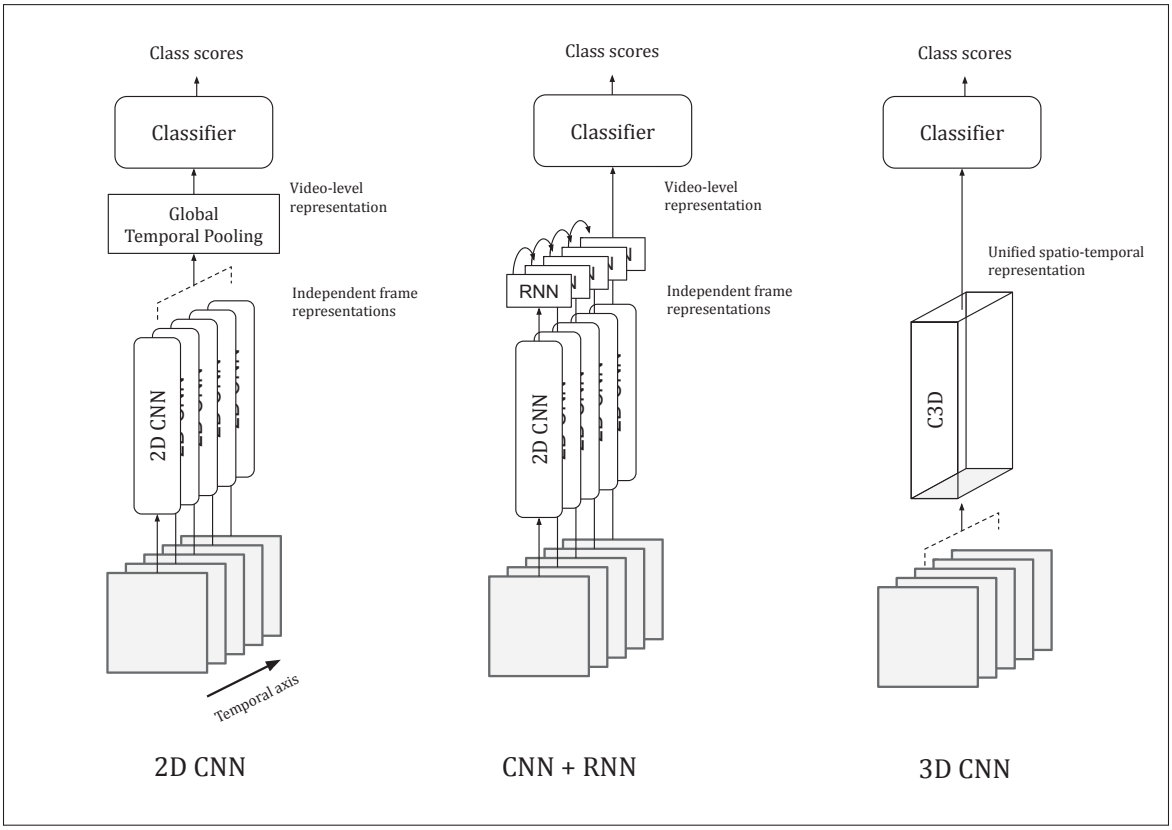

Figure 1.2 Representation of the main approaches for spatiotemporal deep-learning architectures

Recent approaches to video classification rely on deep learning models. Convolutional neural networks (Lecun *et al.*, 1998) are particularly well suited for feature extraction in images and videos, due to the large and varying size of the input. Indeed, the number of learned parameters does not depend on the input size, as the same kernel is used over all locations. Also, they provide spatial or spatiotemporal invariant in the feature extraction.

Figure 1.2 represents the main architecture types for spatiotemporal deep learning with RGB frame sequences. At each layer of an architecture, it is important to understand which type of

process is operated. It can be a simple aggregation of representations or an actual extraction of features, along spatial or temporal axes, and with different abstraction levels of representation.

In deep neural networks, every layer produces a representation of the input, as a set of extracted features. A layer produces a higher-level representation by applying transformations on the preceding layer's representation. Thus the model builds a hierarchy of features, in which abstraction increases with depth.

• **2D CNN:** The CNN extracts a hierarchy of features by detecting spatial patterns from each RGB still-frame independently (Karpathy, Toderici, Shetty, Leung, Sukthankar & Fei-Fei, 2014). The resulting set of frame-level representations is then aggregated to produce a representation of the entire video. Although this approach can produce state-of-the-art accuracy (Bargal, Barsoum, Canton-Ferrer & Zhang, 2016; Vielzeuf *et al.*, 2018), it does not leverage temporal information in the data. Better suited methods have been proposed for video classification.

• **CNN + RNN:** To complement CNN-based architectures, recurrent neural networks (RNNs) are commonly used to recognize temporal patterns in the sequence of high-level frame representations produced by a 2D CNN (Donahue, Hendricks, Guadarrama, Rohrbach, Venugopalan, Darrell & Saenko, 2015; Ng, Hausknecht, Vijayanarasimhan, Vinyals, Monga & Toderici, 2015; Kahou, Michalski, Konda, Memisevic & Pal, 2015; Zhu, Tran, Sevilla-Lara, Yang, Feiszli & Wang, 2020). The implementation of RNNs in combination with CNNs pretrained on large datasets have been widely studied. This type of architecture was first applied to facial expression recognition by Kahou *et al.* (2015). In the work of Vielzeuf (2019), results tend to show that the RNN layer doesn't improve performance over a simple average of the ResNet CNN scores. The author considers data isn't sufficient to train the recurrent layer, which increases the number of parameters of the model. While this two-step, spatial then temporal, approach seems related to the human biological way of processing visual information, it might not be optimal for offline video analysis where the temporal axis of the data and the time of the analysis are distinct. The use of RNNs imposes a constraint on the temporal processing of the image sequence.

Bidirectional RNNs have been studied in order to extract more information by analysing the video in the opposite time direction.

• **3D CNN:** In a more radical way, 3D convolutional networks Tran, Bourdev, Fergus, Torresani & Paluri (2015) unify the temporal and spatial dimensions resulting in a data-volume analysis. 3D CNNs extract a hierarchy of spatiotemporal features from the RGB-frame sequence. Low-level local motion features are extracted in the first layers. The abstraction level and receptive field grow with the depth, up to the last layers recognizing high-level dynamics. The spatial and temporal dimensions are treated as forming a data volume, assuming a similar structure in time and space.

As explained in Karpathy *et al.* (2014) about the Convolutional Neural Networks proposed by Lecun *et al.* (1998), “the spatial structure of images is explicitly taken advantage of for regularization through restricted connectivity between layers (local filters), parameter sharing (convolutions) and special local invariance-building neurons (max pooling)”. If we consider the temporal dimension of videos to have similar structure and properties, spatiotemporal convolutions seem very appropriate. In this report, the term "CNN" will refer to 2D convolutional neural networks, working on static images. Spatiotemporal convolutional networks will be referred to as "3D CNNs".

An important issue with 3D models is their high number of parameters, requiring more data to train and generalize. Results from Carreira & Zisserman (2017) and Hara *et al.* (2018) show that 3D CNNs integrating efficient transfer learning, or simply pretrained on recent large video datasets, perform better than 2D CNNs combined with RNNs. Indeed, the progressive augmentation in available training data was followed by huge improvements in the performance of 3D models on action recognition tasks. In the context of facial expression recognition however, datasets like AFEW (Dhall *et al.*, 2012) are still smaller by an order of magnitude, limiting the performance of 3D CNNs.

The idea of factorized spatiotemporal architectures has been proposed to reduce the complexity of 3D models (Tran, Wang, Torresani, Ray, LeCun & Paluri, 2018). Three dimensional convolution

kernels are split into a 2D spatial kernel and a 1D temporal kernel. An alternation of spatial and temporal convolutional layers replace the stack of 3D convolutions. The resulting (2+1)D method is a compromise between purely spatial approaches and truly spatiotemporal models. These models are easier to train, as the decomposition of kernels facilitates optimization. However, the ability of factorized convolutions to capture spatiotemporal patterns is not clear, and it is especially problematic for low-level dynamics. This is explicitely stated in the work of Sun, Jia, Yeung & Shi (2015), as they do not consider local temporal patterns, or low-level dynamics, relevant: "We note that since the ranks of the 3D kernels constructed by [factorization] are generally lower than those of the general 3D kernels, it appears that we sacrifice the representation power by using the factorized scheme. However, spatio-temporal action patterns in general have a low-rank nature, since feature representations of static appearance of human actions are largely correlated across nearby video frames." So the video is still considered like a stack of frames more than a spatiotemporal volume, and the spatial and temporal analyses are separated.

Leveraging appearance and motion analysis jointly is still an issue, although spatiotemporal methods have been studied for a relatively long time. In 2010, Ji *et al.* (2010) already proposed to apply 3D convolutions on spatiotemporal data volumes extracted around detected human faces. This idea is the basis of the classification method we are working with and has been the focus of a lot of research.

Pointing out the high performance of CNNs for image classification, Karpathy *et al.* (2014) study different approaches for adapting them to activity video classification, varying the temporal connectivity of the convolution, ranging from single-frame CNNs to 3D convolutions. The "surprisingly modest improvement" they obtain by leveraging temporal information compared to purely appearance based methods leads them to conclude that motion cues might be relatively irrelevant to the task. However, their results probably just reveal a failure to capture the temporal evolution in the video data. To facilitate this process, Simonyan & Zisserman (2014) explicitly complement the appearance analysis, augmenting still RGB frames with precomputed optical-flow describing low-level motion between frames. Several works have shown the efficiency of

optical-flow for video classification with CNNs (Ng *et al.*, 2015; Varol, Laptev & Schmid, 2018; Sevilla-Lara, Liao, Güney, Jampani, Geiger & Black, 2018).

Another direction of research with 3D CNNs is the widening of the temporal range of the analysis Varol *et al.* (2018). Indeed, models usually include pooling layers to aggregate the features from different spatial and temporal positions, producing a prediction for the entire input. But the actual range of the temporal analysis is limited, preventing the detection of long term patterns. Also, for training time consideration, videos are often divided into shorter clips. Clips are short temporal windows consisting of several contiguous frames extracted from a video. This can be seen as a form of data-augmentation, but limits the temporal range of learned features. We discuss this issue in Section 1.2.3.

In the general context of action recognition, Sevilla-Lara *et al.* (2019) and Huang, Ramanathan, Mahajan, Torresani, Paluri, Fei-Fei & Niebles (2018) evaluate the importance of motion analysis, and the ability of state-of-the-art models to capture it. A common conclusion is that 3D CNNs still rely heavily on appearance, more than motion, because of the way tasks and datasets are created. Also, they hypothesize that long-term patterns are not necessary for action recognition, but a wide receptive field is important to select the most relevant frames.

Despite their limitations, 3D CNNs constitute the state of the art for video classification, on action recognition especially thanks to the large available training data. The datasets for video facial-expression recognition still lag behind by an order of magnitude or more, limiting the performance of 3D models.

1.2.2 3D CNNs for the EmotiW challenge

This section describes some research related to the EmotiW audio-video challenge (Dhall *et al.*, 2018) and how they motivate our study. To understand the difficulty in leveraging spatiotemporal features with 3D CNNs on the EmotiW challenge, it is interesting to look at the Orange Lab and Valentin Vielzeuf (OL_UC) team's participation history. The winners of the EmotiW 2016 video recognition challenge, Fan, Lu, Li & Liu (2016) used a fusion of CNN+RNN and 3D-CNN

representations to classify videos. The C3D network (from Tran *et al.*, 2015) was pre-trained on the Sports-1M dataset (introduced by Karpathy *et al.*, 2014). In 2017, Vielzeuf *et al.* (2017) inspired from this work and achieved fourth place (Dhall, Goecke, Ghosh, Joshi, Hoey & Gedeon, 2017). They improved the C3D accuracy by developing a weighted C3D. The model assigns weights to the different segments compositing a video, depending on their scores during the training phase, in order to learn more from the meaningful parts of the videos and reduce noise in labels. The same year, the submission of Knyazev, Shvetsov, Efremova & Kuharenko (2018) achieved second place on the challenge, without using any temporal feature extraction on the frame sequences. In 2018, this direction was followed by the OL_UC team (Vielzeuf *et al.*, 2018) who made a point of not using any motion features for simplicity and efficiency. They obtained third place focusing only on spatial information, aggregating frame-level representations to classify videos Dhall *et al.* (2018).

Literature results reported in Table 1.1 cannot be compared directly due to heterogeneous experimental setups (not always detailed). They describe validation performance of models on the image data only (no audio), usually provided in ablation studies evaluating larger model ensembles that achieved top-3 performance on EmotiW audio-video challenges (AV). It appears that best performing methods on AFEW are built around 2D CNNs combined with recurrent networks. Simpler models with no temporal features also perform well. 3D models lack in performance on this task, probably because of the limited training data. Combining models, with fusion of features or class predictions, is very popular as the resulting model ensemble benefits from the diversity in expertise of each component (Liu *et al.*, 2018a; Fan *et al.*, 2016). It has also be shown that leveraging external training data can greatly improve the model's performance, even with 2D CNNs (Liu *et al.*, 2018a; Bargal *et al.*, 2016).

This shows how spatiotemporal cues can be tricky to learn for the AFEW dataset. We gave some arguments, in Section 1.1.1, showing that dynamics still have an important role to play in video FER. Our study aims at finding, through literature review and experiments, how 3D CNNs can overtake the obstacles to efficient spatiotemporal analysis in FER, and perform as well as they do on action recognition, where they push the state of the art higher.

Table 1.1 Heterogeneous results reported on the AFEW dataset for validation of model or ensemble components, for top-tier submissions on the audio-video EmotiW challenge

Submission reference	**Component**	**Acc. (%)**
Li, Zheng, Zong, Lu, Tang, Jiang, Liu & Xia (2019) 2nd place AV EMOTIW 2019	ResNet-18 + BLSTM DenseNet-121 + BLSTM	43.34 49.35
Lu, Zheng, Li, Tang, Liu, Yan & Zong (2018) 3rd place AV EMOTIW 2018	3D VGG-16 VGG + BLSTM	39.36 53.91
Liu *et al.* (2018a) 1st place AV EMOTIW 2018	4CNNs + LSTM 3D (LMED) + SVM	56.13 39.95
Fan, Lam & Li (2018) 2nd place AV EMOTIW 2018	VGG-Face FG-Net	45.16 47.00
Vielzeuf *et al.* (2018) 3rd place AV EMOTIW 2018	ResNet-18 av. pool. weighted av. pool	49.7 50.2
Vielzeuf *et al.* (2017) 4th place AV EMOTIW 2017	C3D + LSTM Weighted C3D	43.2 42.1
Fan *et al.* (2016) 1st place AV EMOTIW 2016	C3D 2 CNN-RNNs + 2 C3Ds	39.69 48.30
Bargal *et al.* (2016) 2nd place AV EMOTIW 2016	VGG-13 + STAT + SVM ResNet + STAT + SVM	58.9 52.62

1.2.3 Temporal receptive field and sample duration homogenization

Sample videos from the AFEW dataset have varying length (Dhall *et al.*, 2012), similarly to some major datasets for general action recognition like Sports-1m (Karpathy *et al.*, 2014) or Kinetics (Kay, Carreira, Simonyan, Zhang, Hillier, Vijayanarasimhan, Viola, Green, Back, Natsev, Suleyman & Zisserman, 2017). In order to perform efficient and fast training, it is important to compose batches of samples with fixed length (e.g. for batch normalization, parallel computation). In the context of image classification, the common approach is to resize inputs to a fixed spatial resolution, in order to create uniform batches of data for training, and to allow the network to produce a fixed-size output score, with some level of scale invariance. For video classification however, with the additional temporal dimension, the uniformization of data is more complicated. In a given application or dataset, frame sequences can vary consequently.

The main approach to sample duration homogenization is to extract one or several fixed-length temporal windows from the samples. This training technique is present in Baccouche, Mamalet, Wolf, Garcia & Baskurt (2011); Tran *et al.* (2015) and Carreira & Zisserman (2017) among others. In the following subsections, we will discuss two issues related to this technique:

– the model is trained on short sequences and cannot learn long-term patterns;
– the video-level ground-truth labels do not necessarily correspond to the content of shorter clips.

1.2.3.1 Learning long-range dependencies

Varol *et al.* (2018) note that most of the research work on spatiotemporal 3D CNNs considers very short video intervals (2 to 16 frames). However, we have to differentiate the low-level kernel size, the model's global receptive field and the size of training clips, and also consider temporal downsampling which influences the network's capacity to model motion features.

Taylor, Fergus, LeCun & Bregler (2010) use a convolutional gated restricted Botlzman machine (GRBM) to extract local features from every successive pair of frames. Then, a 3D convolutional layer captures mid-level spatiotemporal cues, combining features from different positions with kernel size $9 \times 9 \times 9$. Ji *et al.* (2010) extract spatiotemporal cubes around detected actions from 7 consecutive frames with a step of 2 frames. They use 3D convolution kernels with size 3 in the temporal dimension. Karpathy *et al.* (2014) use training clips of 10 frames (less than half-second videos) for spatiotemporal convolutions. Another of their architectures works with only 2 frames, separated by a 15-frame interval. This corresponds to a receptive field of 2 frames after 1/15 downsampling. Tran *et al.* (2015) randomly select clips of 16 consecutive frames for training. During the testing phase (or deployment), batches are not used so the convolution is carried out on longer samples, and the prediction corresponds to the most frequently occurring label. This practice is supported by the work of Schindler & Van Gool (2008), showing that short snippets of 7 frames (at 25 fps) are enough to classify action videos. The authors provide observations from biological studies to back up this idea. However, the study focuses on simple and controlled action video datasets.

To deal with more realistic data, Ng *et al.* (2015) evaluate the benefits of learning a global description of the video's temporal evolution. Instead of aggregating clip-level descriptions to predict a video label, they implement a recurrent network whose hidden state evolves while processing the frame sequence. The 3D-CNN paradigm limits the duration of videos that can be processed, because of its fixed receptive field and its computational weight. They study the CNN with LSTM method (Hochreiter & Schmidhuber, 1997), using RGB and optical-flow inputs. They trade the hierarchical, implicit, spatiotemporal feature extraction of 3D convolutions for a longer-range temporal pattern recognition, with explicit local motion information. Their model is trained on radically longer sequences, with 120 frames at 1 fps (2 minutes from more than 5 minutes real-world action videos). Training with longer video clips also requires to pad or loop samples that do not have enough frames. The good performance of their models shows the advantage of using the entire video rather than short clips during training.

With the same objective of leveraging the long-term temporal structure of action videos, Varol *et al.* (2018) directly increase the global temporal receptive field of the 3D architecture. To keep the computational cost and memory usage manageable, they reduce the spatial resolution of the input, down to 58×58 when increasing the duration. This way, they obtain better performance with and without optical-flow, using clips of 64 or even 100 frames.

This technique consisting in learning from small clips is still widely used for 3D CNNs in recent literature, with 8 frames in Hou, Chen & Shah (2017) and 16 frames in Hara *et al.* (2018). The main reason is the need of uniform data batches with limited memory, for faster training and regularization with batch-normalization.

However, Carreira & Zisserman (2017) also explain that using lighter models allows the use of spatially and temporally larger clips for training, which can improve classification accuracy. This is not obvious in their experiments because the improvement can come from the difference in depth, parameter number, pre-training (from 2D ImageNet data), spatial resolution or temporal footprint of the models.

An interesting balance between using short clips for efficient computation and long sequences for better temporal representation is the early work of Hou *et al.* (2017) who used a RNN to capture evolution in the clip-level features extracted by a separately-trained 3D CNN. The model is not trained end-to-end however.

In the experiments of Tran *et al.* (2018), the optimal clip-length is 32 frames. With longer clips, the clip-level prediction accuracy continues improving, but the video-level classification accuracy decreases. For efficient training, they propose to pre-train the model with small clips (which is faster), and to finally fine-tune it with 32-frame clips (which is more accurate). Note that 32 frames remain a very limited temporal windows, usually corresponding to 1 or 2 seconds of video.

Instead of increasing the duration of video clips used for training and testing with a convolutional architecture, Diba, Fayyaz, Sharma, Karami, Arzani, Yousefzadeh & Van Gool (2017) work on increasing the receptive fields of convolution kernels to capture more interesting spatiotemporal patterns. They propose a layer using kernels of different temporal depths, inspired by the Inception network (Ioffe & Szegedy, 2015). They also use 32-frame clips.

Even with 3D models trained on short clips only, it is still interesting to analyse the entire videos during the inference phase, and this is usually possible as gradients are not computed and batches are not used. The clip-level features can be aggregated to produce the global video description. This requires a length-agnostic model that can take arbitrary duration videos as input, with a recurrence mechanism or in a simpler way with temporal pooling.

1.2.3.2 Clip extraction for 3D CNNs

If the duration of clips is an important research topic, the process of selecting their position in the sequence is another interesting issue related to the temporal data-sampling required to build training inputs for 3D CNNs. Different ways of extracting clips from the video examples are presented in Figure 1.3.

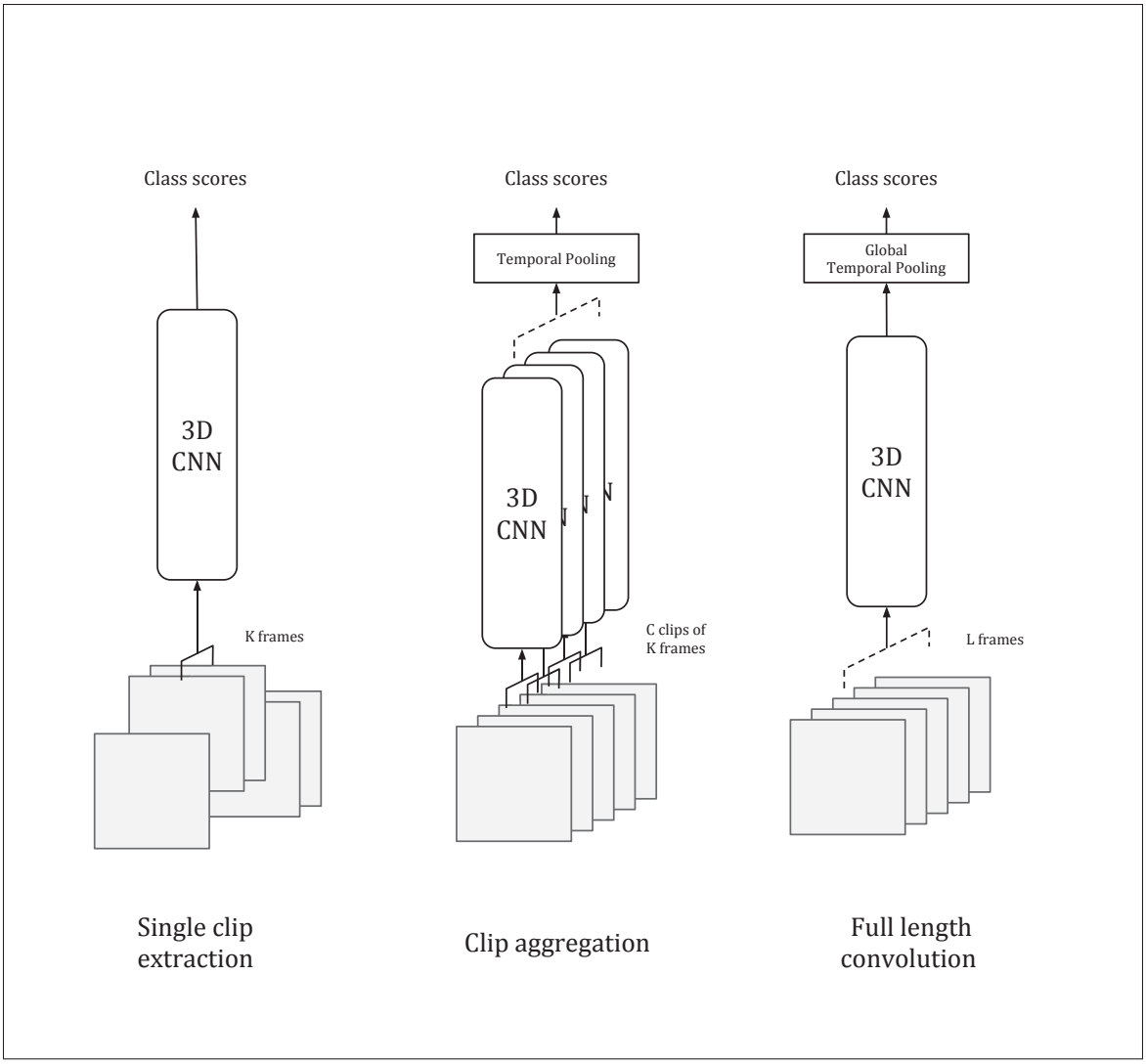

Figure 1.3 Clip-extraction modalities for training and testing. K and C are constants fixed to obtain uniform batches. L is the arbitrary length of the entire video sequences. Global pooling allows to aggregate these arbitrary-length representations, as described in Section 1.2.4.1

• **Single-clip extraction:** A simple technique to obtain uniform training batches is to extract a fixed-length clip from each video sample. Extracting this clip randomly, at a different starting frame for every training epoch, adds some form of data-augmentation. Since the emotional cues are not distributed uniformly across the frames, the video-level labels are less accurate to describe extracted clips. Temporal windows only contain a fraction of the relevant spatiotemporal features from the entire sample. Thus, using only a random portion of each sample will induce

noise in the training labels. It is usually considered that the benefits of having uniform batches overtake this issue, and the noise is reduced through iterations with random clip locations. Also, the location of the features are lost when the clip is extracted from the video. This can be seen as a Multiple Instance Learning (MIL) task, using the bag-of-word representation as described for video classification by Laptev, Marszalek, Schmid & Rozenfeld (2008) and Niebles, Wang & Fei-Fei (2008). Each video can be considered as a bag of temporal windows, with one single label.

A lot of research work use single-clip extraction to build uniform inputs for the training phase. For action recognition, Tran *et al.* (2015), Hara *et al.* (2018) and Carreira & Zisserman (2017) randomly extract a single 16-frame clip form each video sample for training. On the 2016 EmotiW challenge, Fan *et al.* (2016) used the same strategy.

• **Clip aggregation:** During testing, there is no need for uniform inputs, so the test samples are split into smaller clips that can be fed to the model. For each test video, Tran *et al.* (2015) and Fan *et al.* (2016) respectively use 10 and 16 clips of 16 frames and averaged the resulting scores. Hara *et al.* (2018) propose a similar strategy with the number of clips depending on the video duration. In the context of action detection and segmentation, Hou *et al.* (2017) also use clip aggregation for testing.

Clip aggregation can also be used directly for training. Vielzeuf *et al.* (2017) developed a weighted C3D, training with several clips and assigning them weights to determine which clips are the more relevant to learn from. This strategy should be able to reduce the noise in training labels induced by clip extraction. Liu, Liu, Gan, Tan & Ma (2018b) also uses several clips to combine their features into a video-level representation, with a learned weighted pooling.

• **Full-length convolution:** At test time, the most satisfying prediction technique is to use as much clips as possible from the video, to take into account all available information. With fully-convolutional architectures, it is possible to convolve the model temporally over the entire video, in a sliding-window manner. This is equivalent to feeding every possible k-frame clips to the network, where k is the size of the model's temporal receptive field. The model

produces features or class scores independently at every time step. This output can be temporally aggregated as usual to obtain the video-level representation. However, this doesn't allow models to capture long-range patterns, as they cannot learn this behavior during the training phase with short clips. This method is used in the state-of-the-art work of Carreira & Zisserman (2017). The training is performed with random single-clip extraction, and the tests are done by applying the model convolutionally on the entire videos, averaging clip scores.

1.2.4 Temporal pooling

"The idea of feature pooling originates in Hubel and Wiesel's seminal work on complex cells in the visual cortex (1962), and is related to Koenderink's concept of locally orderless images (1999)", Boureau, Ponce & LeCun (2010).

As stated by Boureau *et al.* (2010), feature pooling is widely used to obtain some level of invariance to image transformation and robustness to noise and clutter. Since it consists in aggregating feature vectors from different positions, it also provides more compact representations. Practically, temporal pooling allows us to obtain a global video description from a set of clip-level feature vectors.

Temporal feature pooling has been used extensively for video recognition as explained by Ng *et al.* (2015). Notably, in the context of bag-of-words representation, as described in Laptev *et al.* (2008), Wang, Ullah, Kläser, Laptev & Schmid (2009) or Sivic & Zisserman (2003), spatiotemporal features are extracted from different locations in the videos (densely or at interest points). These descriptors are quantized into a dictionary built with a clustering technique (e.g. k-means) and then aggregated across the space and time domains to obtain the bag-of-word representation. Interestingly, the same aggregation method is used along the temporal and spatial axes (a video can be represented by the frequency histogram over the dictionary features, like a bag of features).

At clip level, the sliding-window feature extraction inherent to 3D convolution is similar to this idea, building a representation by summing the correlation scores of all input locations with the filter. At video-level, temporal pooling allows us to further aggregate features.

1.2.4.1 Fully convolutional architectures with global pooling

Global average pooling (GAP) was proposed in Lin, Chen & Yan (2013) to replace the fully-connected classification layers usually used at the end of a CNN architecture. Instead of flattening and concatenating the feature maps to feed as input to dense layers producing class scores, the idea of global pooling is to compute the average of each feature map across positions.

Considering a vector F of C spatiotemporal feature maps, with dimensions (L, H, W), global average pooling can be applied as follows:

$$GAP(F_c) = \frac{\sum_{k=0}^{L} \sum_{m=0}^{H} \sum_{n=0}^{W} F_{c,k,m,n}}{L \times H \times W} \qquad \forall\ c < C. \tag{1.1}$$

The resulting vector is fed directly into a softmax layer and interpreted as class scores. This enforces a direct correspondence between convolution filters and categories. The averaging of the feature maps along the spatial and temporal dimensions makes this method length agnostic and robust to translations of the input. The authors also emphasize its regularizing property. Indeed, fully connected layers are prone to overfitting because they concentrate a very high number (usually more than half) of trained parameters in the last layers of the networks. Also, cross-channel convolutions with $1 \times 1 \times 1$ kernel-size can replace fully-connected layers. This type of layer operates a transformation of the features at every location in the 3D maps, with shared parameters, and can be used to reduce the feature-space dimensionality.

1.2.4.2 Pooling position

Pooling layers can be placed at different levels in the network architecture. The comparison of feature-level pooling and score-level pooling is interesting but the delimitation is not very

clear. In deep-learning architectures, the layers extract a hierarchy of features, the most abstract, high-level, features being considered as class scores. All preceding layer outputs can be considered as features. The delimitation between a feature extractor and a classifier is usually more practical than theoretically-based. In the case of feature-level temporal pooling, frame representations are aggregated to produce a set of feature-maps or feature-vectors describing the entire video. On the opposite, with score-level temporal pooling, the classifier provides a score for each position along the temporal axis that will be aggregated. Figure 1.4 represents both modalities. Different combinations can be considered, with feature-level and score-level pooling for spatial and temporal dimensions.

Karpathy *et al.* (2014) compare temporal fusion modalities placed at different levels in the architecture. They fuse frame information by implementing temporal connectivity in the convolutional layers, without using an actual pooling mechanism. They find that their *slow fusion* is the most efficient, progressively aggregating frame information from larger windows throughout the network, as it is usually done with spatial positions within images. However, they do not keep the global receptive field of the network constant across the experiments so the results can't be interpreted easily. They also show that operating spatial and temporal convolutions works better than stacking frames in the channel dimension before applying spatial convolution (which should allow the model to find local spatiotemporal patterns). This could be expected because the weight sharing inherent to convolutional layers offers known advantages. Dissociating channel and temporal dimensions reduces the complexity with a more structured operation.

Ng *et al.* (2015) give a more detailed comparison of *Conv Pooling* (feature-level temporal aggregation), *Late Pooling* (score-level aggregation), *Slow Pooling* (progressive aggregation) and *Time-Domain Convolution*. They work with standard CNN architectures, considering convolutional layers as the feature-extractor part, and fully-connected layers as the classifier. Feature-level temporal pooling performs better than late pooling. Their interpretation is that the spatial information of feature maps has to be preserved for the aggregation of frame information. Another interpretation could be that the feature aggregation is more insightful because the set of

frame-level feature-maps offers better video representation than the set of class scores. With the pooling layer before the classification layer, aggregated features can be oriented toward a bag-of-features representation that is more efficient. On the contrary, score pooling can be seen as a simple counting of the most represented class across the frames. This is especially true in the context of FER with only seven classes, score aggregation would work with even less information (score vectors can be seen as feature maps with one position and 7 channels).

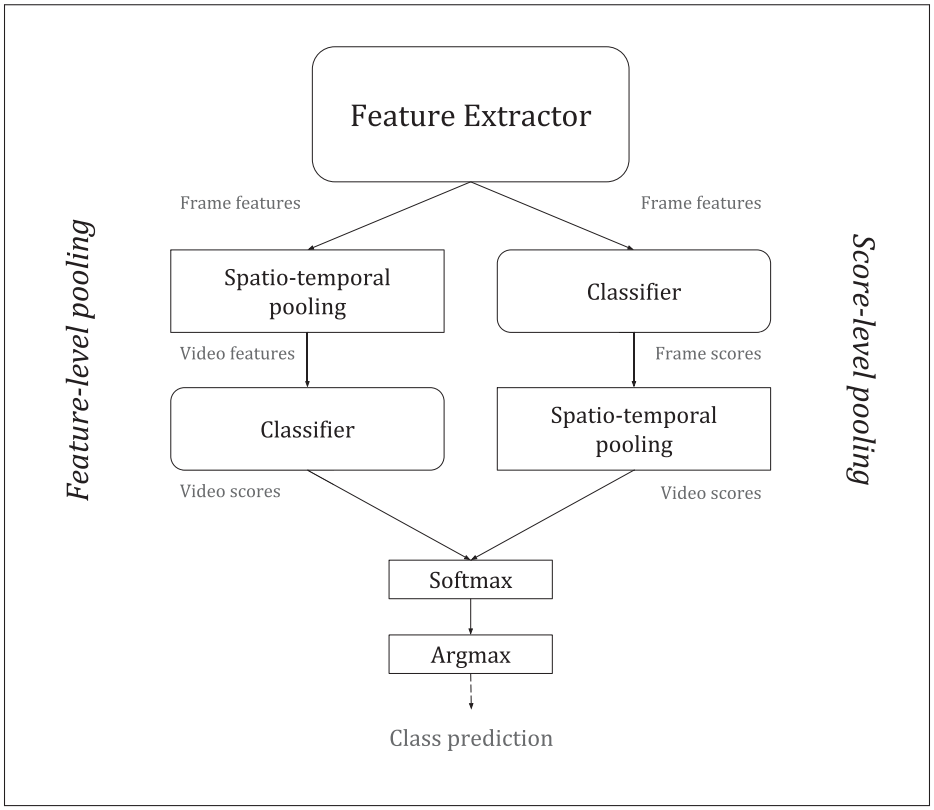

Figure 1.4 Score-level and feature-level pooling modalities

1.2.4.3 Pooling formulation

The most common aggregation methods are *max* and *mean* pooling. Ng *et al.* (2015) consider max pooling more efficient than average pooling because max pooling produces sparse gradients, making the training faster. On the contrary, McFee, Salamon & Bello (2018) explain that it is

harder to train with the rare gradients given by the *max* operator, so they motivate the use of softmax pooling to simulate max pooling with more gradients.

The theoretical analysis of Boureau *et al.* (2010) shows that max pooling works well on features that are sparse across samples, with low probability of activation. They propose to apply pooling in a hierarchical way, pooling over smaller groups of samples and averaging the results. Similar ideas were studied in Karpathy *et al.* (2014) and Ng *et al.* (2015) and performed well.

We can also note that Zhou, Khosla, Lapedriza, Oliva & Torralba (2016) consider that max pooling only encourages the network to recognize a discriminative part while average pooling encourages the identification of the entire object. Their experiments show that max pooling works better for classification and average pooling is more adapted to localization tasks.

McFee *et al.* (2018) study the mean, max and adaptive-softmax temporal pooling in the context of audio multiple instance learning (MIL). Their results on the URBAN-SED database show that max pooling is more adapted to classes in which events are highly localized in time (gun shot, car horn), while average pooling represents successfully events that span across the entire clip range. They confirm the theoretical analysis of Boureau *et al.* (2010) showing that max pooling and average pooling have different ranges of expertise, depending on the input size and the sparsity of features.

Springenberg, Dosovitskiy, Brox & Riedmiller (2015) extend the idea of fully-convolutional architecture by replacing pooling layers with strided convolutions. This is equivalent to learning the pooling operation as a weighted sum in which the weights depend on the relative spatial positions, and not on the feature scores as in previously presented mechanisms. We can consider that this design reduces computational cost but increases the number of trained parameters, so it doesn't go in the direction we favour, because of limited training data.

1.2.5 Transfer learning

Because of the difficulty to learn efficient spatiotemporal patterns from the available video datasets, it is interesting to explore ways of leveraging auxiliary knowledge to help the training process. These approaches are referred to as "transfer learning".

Several approaches have been proposed for transfer learning, including pre-training, 3D kernel inflation and knowledge distillation.

A popular method consists in pretraining models on larger related (and usually more general) datasets. For 2D image classification, the ImageNet database of general object recognition is very popular as a way of pre-training models (Deng, Dong, Socher, Li, Li & Fei-Fei, 2009).

In the case of video FER, human action recognition databases can be used, e.g. Sports-1m and more recently Kinetics. The idea is that the large quantity of labelled data enables the learning of spatiotemporal patterns that will be reused on the more specific final task. As the task gap between object recognition and FER is quite large, more related tasks can be used. Frame-based models for FER can benefit from pretraining face recognition datasets, as VGG-Face, with better results than with ImageNet (Kaya, Gürpinar & Salah, 2017; Knyazev, Shvetsov, Efremova & Kuharenko, 2017). There is no straight-forward equivalent video dataset for spatiotemporal pretraining.

With the I3D model, Carreira & Zisserman (2017) propose an inflation strategy consisting in bootstrapping the training process with 2D image classification data. The transferred knowledge will be oriented toward a more closely related task but without incorporating the temporal analysis. This strategy seems perfectly relevant considering the already impressive performance of 2D CNNs on video classification tasks. This knowledge transfer is achieved by expanding pre-trained 2D convolution kernels into 3D kernels, by replicating them along the temporal dimension. The new weight values have to be divided by the temporal kernel size to preserve activations. Fine-tuning this new bootstrapped 3D model with video data enables learning of spatiotemporal features.

Highlighting the difficulty to interpret their results, the authors state: “The difference over the C3D-like model can be explained by our I3D models being much deeper, while having much fewer parameters, by leveraging an ImageNet warm-start, by being trained on 4× longer videos, and by operating on 2× higher spatial resolution videos”, (Carreira & Zisserman, 2017).

In the context of FER however, the impact of the task gap between human action classification and emotion recognition for the pre-training is still unclear. Models can be pre-trained on more related tasks as 2D face recognition, or directly on 2D emotion recognition, but the size of available datasets usually decrease with their specialization.

Another interesting question concerns the actual importance given to temporality in inflated models. Specifically, it is difficult to know if the model uses the transferred 2D knowledge to learn new spatiotemporal dependencies, or simply focuses on the spatial aspect. Sevilla-Lara *et al.* (2019) show that inflated models are biased towards motion-independent classes. The strong 2D pre-training allows good recognition of spatial features but doesn’t favour learning temporal patterns. They also explain that current video datasets are typically built in such ways that temporal analysis only has limited benefits. This has been discussed for the AFEW dataset in Vielzeuf *et al.* (2018). Using such spatially biased datasets for benchmarking will favour models that give more importance to appearance.

Sevilla-Lara *et al.* (2019) and Huang *et al.* (2018) propose methods to evaluate the importance of motion analysis in a model. We can also note that methods have been studied to enforce temporal awareness in the network. Misra, Zitnick & Hebert (2016) use the frame-order recognition as a self-supervised, auxiliary task to train the network.

Knowledge distillation (Hinton, Vinyals & Dean, 2015) is another interesting transfer-learning approach, based on using separate architectures for the training and the deployment stages. We do not investigate it in this work.

1.2.6 Spatio-temporal attention

Attention is another way of improving a deep-learning model's performance, by modifying its architecture to incorporate a specific module that will not directly work in the feature extraction pipeline, but rather assist it.

Attention mechanisms consider the context of each feature in order to determine their relevance for the representation of the specific input data sample. Highlighting this relation between the concepts of attention and context, Britz, Goldie, Luong & Le (2017) state that the *attention vector* is another name for the *context vector*, in their framework of neural machine translation (NMT). The idea is to select features to highlight or inhibit, for a specific situation summarized in the context vector computed from the input. To this end, an attention weight is attributed to each feature, or entity. This behavior contrasts with usual convolutional kernels which learn data-driven patterns that are generally relevant to the task, without inferring their importance from other features in the input.

Although some attention mechanisms are explicitly designed for vision tasks (Mnih, Heess, Graves & Kavukcuoglu, 2014; Jaderberg, Simonyan, Zisserman & Kavukcuoglu, 2015), a lot of ideas originate from neural machine translation as a way to perform alignment (Bahdanau, Cho & Bengio, 2015; Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser & Polosukhin, 2017). Alignment in language translation is a different concept than what we use for facial alignment. Face alignment consists in cropping and morphing the faces to the same canonical spatial pose. In neural machine translation, words or tokens produced for the output sentence are aligned to relevant tokens from the input sequence. This allows efficient translation since the source and target languages have different grammars and ordering of words. For instance, the verb in the target language can have a different position than the verb in the source sequence, but its representation has to be determined from the related verb, subject and tense cues from the input sentence in order to obtain desired meaning and conjugation.

These attention mechanisms can be adapted to vision tasks by considering tokens as pixels instead of words (Xu, Ba, Kiros, Cho, Courville, Salakhutdinov, Zemel & Bengio, 2015; Parmar,

Vaswani, Uszkoreit, Kaiser, Shazeer, Ku & Tran, 2018). Differences of dimensionality and size of the representations have implications on the way the computation has to be adapted.

Typical attention mechanisms produce attention weights for all positions. Various mechanisms and behaviors can be developped based on this principle. Notably, attention weights can either be used as aggregation weights in a pooling operation or as an attention mask before a feature-extraction layer. An interesting criterion to compare attention mechanisms is the type of weighting they operate. Attentional behaviors can be identified as hard or soft.

1.2.6.1 Soft attention

In soft-attention mechanisms, attention weights determine the importance given to each entity in the input (spatial or temporal positions, or words in a sentence). A new representation is computed by weighting the features. Notable examples include Bahdanau *et al.* (2015); Xu *et al.* (2015); Vaswani *et al.* (2017).

A real-value attention mask can be produced by regressing an attention weight for each position in the input features, using a learned transformation (e.g. convolutional layers).

Context information can be leveraged to produce the attention score for each position. Xu *et al.* (2015) implement soft spatial attention in a sequential generation task for captioning. Attention over spatial locations in the image does not only depend on the features present at each locations, but is also conditioned on the hidden state of the LSTM for the current generation step (encoding context information of the entire image and also information of what has been produced before, and what word is needed at this step).

Interestingly, some research work skips this idea of context, and use purely feature-dependent attention weights. Li, Bak, Carr & Wang (2018a) employ a two steps attention, with spatial attention first and then temporal attention. Woo, Park, Lee & Kweon (2018) apply attention on the channel dimension followed by the spatial dimension.

On the context of emotion recognition, Lee, Kim, Kim & Sohn (2018) use a spatial attention mask for each frame in a video. Kumar, Rao & Yu (2020) produce a temporal attention profile over the frame sequence. And both temporal and spatial attention are combined in Aminbeidokhti, Pedersoli, Cardinal & Granger (2019).

The Transformer designed by Vaswani *et al.* (2017) popularized the concept of self-attention, weighting positions with a compatibility function (that can be implemented with a dot product) between pairs of features. For each position (or entity), an attention profile over all positions is used for a weighted sum. In the Transformer architecture, attention and feature extraction are intertwined, removing the need for convolutional or recurrent layers. This was further studied in Wang, Girshick, Gupta & He (2018b) with the development of Non-Local Neural Networks. However, self-attention can also easily be introduced into a CNN to model attention (Zambaldi, Raposo, Santoro, Bapst, Li, Babuschkin, Tuyls, Reichert, Lillicrap, Lockhart, Shanahan, Langston, Pascanu, Botvinick, Vinyals & Battaglia, 2019).

1.2.6.2 Hard attention

Hard attention can be seen as a special case of weighted attention in which weights are either 0 or 1 (Xu *et al.*, 2015). In practice, sampling mechanisms allow to reduce complexity and save computation by focusing feature extraction only on the region of interest (ROI). This behavior is studied in Mnih *et al.* (2014); Jaderberg *et al.* (2015); Korbar, Tran & Torresani (2019).

The Recurrent Attention Model (RAM) proposed in Mnih *et al.* (2014) and further studied by Ba, Mnih & Kavukcuoglu (2015) and Sermanet, Frome & Real (2015) is based on the human visual attention, gathering information from different locations by orienting the fovea to salient objects. A single image is processed by a sequential model, accumulating information from various locations through a RNN. This idea of processing an image as a sequence of glimpses was already studied by Larochelle & Hinton (2010). This mechanism implements hard attention as it only focuses on part of the input, while other positions will not be processed. Because of the non-differentiable, stochastic nature of this sampling mechanism, RAM is trained with

reinforcement learning which makes it harder to use, as stochastic gradient estimates can have high variance. This explains the limited popularity it received in literature, other than work to improve the training method.

Other sampling mechanisms can implement differentiable hard attention. The Spatial Transformer Network (Jaderberg *et al.*, 2015) uses a parameterized affine transformation to crop (but also scale, rotate, shear...) the image. Similarly, the selective attention model of the Deep Recurrent Attention Writer (DRAW) uses a grid of Gaussian filters to sample an image region and transform it (Gregor, Danihelka, Graves, Rezende & Wierstra, 2015). This mechanism implements a soft weighting of pixel values with Gaussian distributions, making it differentiable, but it produces hard attention behavior, working on a small patch of spatially sampled data. In these models, hard attention is implemented with differentiable sampling mechanisms. The modules produce parameters for the spatial transformations that operate cropping.

On the contrary, the Gated Attention Network (Xue, Li & Zhang, 2020) introduces a hard-attention mechanism with binary weights assigned to each position. This allows for non-localized attention, coupled with the benefit of saving computations as in hard-attention. With binary attention weights, the model is not directly differentiable, so the authors use a Gumbel-Softmax relaxation to make it differentiable (to avoid reinforcement learning).

In Chapter 4, we discuss and experiment typical attention mechanisms that can be applied to deep-learning models. We consider the benefits and challenges for their implementation in a spatiotemporal analysis framework.

CHAPTER 2

EVALUATION OF DEEP SPATIO-TEMPORAL ARCHITECTURES FOR FACIAL EXPRESSION RECOGNITION

This chapter discusses architectural designs for 3D CNNs. We evaluate two popular video-recognition models with different pretraining methods. Section 2.3 provides ablation studies to characterize the impact of architectural configurations such as the position and behavior of the spatial and temporal aggregation mechanism, or video duration homogenization and clip size characterization. The discussion provides interpretations for the relative performance of the models, considering the specificities of the task, emotion recognition in videos.

2.1 Architectures under evaluation

2.1.1 Pre-trained 3D models

We fine-tune two pre-trained 3D models from Hara *et al.* (2018)[1] and replace their final classification layer by a new one trained from scratch for 7 classes emotion recognition. The ResNet-18-kinetics is the simpler model, proposed by He, Zhang, Ren & Sun (2016), with 8 ResNet blocks (18 convolutions on the direct path). The architecture contains a total of 17 3D convolutional layers with 33M parameters. The model produces a vector of 512 3D feature maps describing the video clip that we can feed to a classifier.

The ResNeXt-101-Kinetics architecture is based on the ResNeXt module from Xie, Girshick, Dollár, Tu & He (2017) which is a modification of the ResNet. The ResNeXt blocks are composed of three convolutional layers with a residual connection on the entire blocks. A convolution with kernel $1 \times 1 \times 1$ on the residual path allows downsampling and dimensionality changes in the residual features to match the output of the block, when required. The architecture contains a total of 104 3D convolutional layers (100 convolutions in the main path), with 47M parameters. Even though the ResNeXt-101 architecture is much deeper and produces more

[1] https://github.com/kenshohara/3D-ResNets-PyTorch

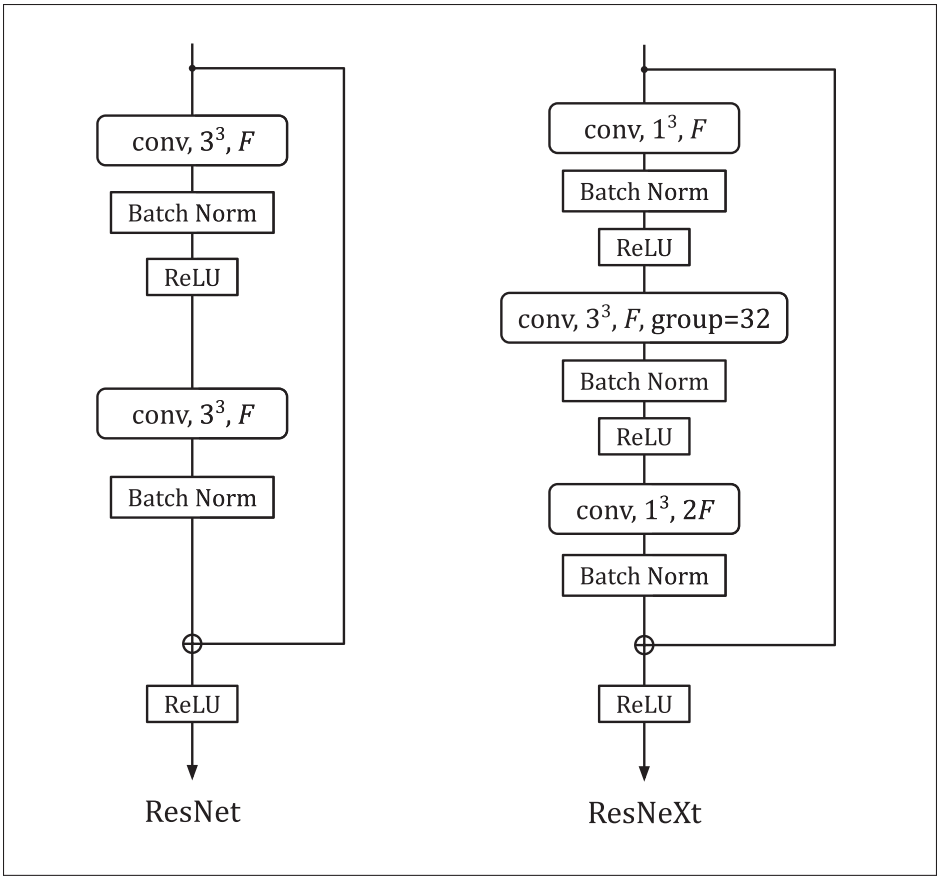

Figure 2.1 Basic blocks composing the 3D-CNN architectures from Hara *et al.* (2018). The notation *conv,* x^3*,* F represents the kernel size $x \times x \times x$, and the number of feature maps F of the convolutional filter. The ResNeXt block also uses group convolutions, separating feature maps by groups of 32. Shortcut connections use summation

features maps than the ResNet-18, the grouped convolutions and its bottleneck design limit the growth of the parameter number.

Both models have been trained on the Kinetics dataset with over 300K human action videos (Kay *et al.*, 2017). The block and model architectures are illustrated in Figure 2.1 and Appendix I-1. Convolutional layers in the models do not use bias weights. Since the ResNet-18 models requires less memory to train, we can remove temporal stride in all of its convolutional layers. This allows it to produce a temporally dense output, preserving the number of frames in the temporal dimension of the output features.

2.1.2 Spatio-temporal classification

Because the pretrained models produce features that do not directly correspond to our problem, we complement the 3D ResNeXt-101 or ResNet-18 networks with a final 3D classification layer with 7 output classes. The classification layer is a 3D convolution with kernel $1 \times 1 \times 1$ on the 2048 or 512 feature maps. We primarily use this design for its capability of working with arbitrary durations of video input.

As discussed in section 1.2.4.2, we compare feature-level and score-level pooling. With deep-learning, this separation is abstract and conceptually unclear as we work with features at different degrees of abstraction. For convenience, we consider the pretrained 3D ResNet-18 architecture to be the feature-extractor, and the extra layer(s) that we add is the classifier.

We compare 3 pooling modalities detailed in Figure 2.2:

– 3D global average pooling at feature level,
– 3D global average pooling at score level,
– 2-step pooling, with spatial max pooling at feature level and temporal average pooling on the sequence of scores.

This 2-step pooling strategy is simply a formulation of what is commonly found in the literature, with spatial max-pooling throughout the CNN architecture, between convolution layers, and a temporal aggregation of the temporal scores obtained for all frames, or clips.

2.1.3 Inflated 3D model

We use a 2D VGG-16 model (Simonyan & Zisserman, 2015), pretrained with VGG-Face and FER2013 image datasets[2], and inflate it to a 3D model as described in Section 1.2.5.

When inflating a 2D CNN in the temporal domain, several questions arise concerning the details of the new architecture to build, and the training procedure to use. The inflated network is not

[2] https://github.com/XiaoYee/emotion_classification

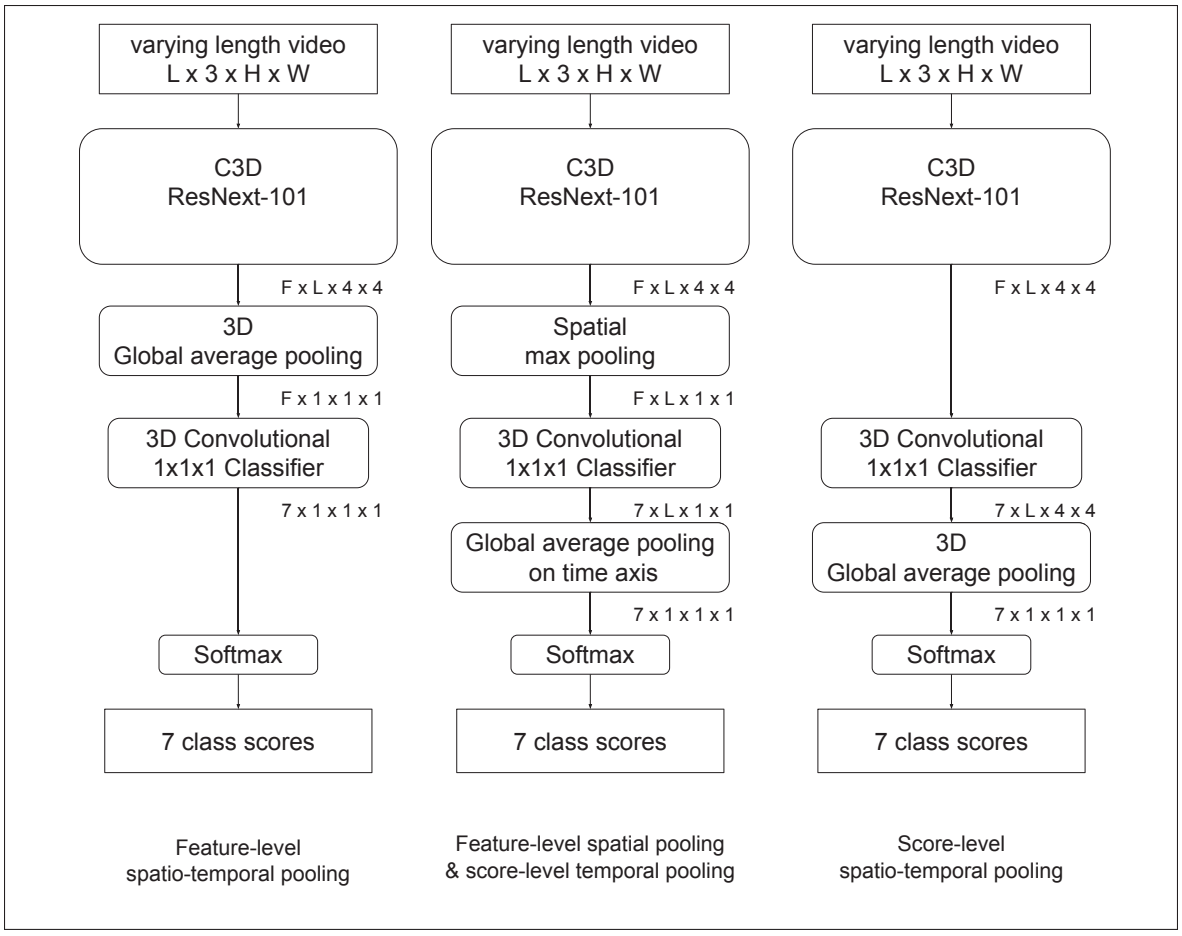

Figure 2.2 Different modalities of global average pooling we use for experiments with fully 3D convolutional architecture. L is the length of the frame sequence, fixed during training and the full duration of samples for test time. $H \times W$ is the spatial resolution of inputs, 112×112 for our experiments. C is the number of input data channels, there are 3 colour channels for RGB data. F is the number of 3D feature maps, 2048 at the output of ResNeXt-101, 512 for ResNet-18

pre-trained with any particular temporal window size or clip-extraction strategy so the results are not biased by pre-training choices. This allows us to find the best suited approach for the task.

The authors Carreira & Zisserman (2017) experiment the i3D strategy with an inception network, incorporating different kernel sizes in both the temporal and spatial domains. We use a VGG network with 3×3×3 kernels, even though we could chose different temporal sizes. This is a

common approach because it limits the number of parameters in each layer and thus allows to build deeper networks.

Kernel temporal size is 3 in every layer, with no temporal pooling inside the network. The model contains 164M trained parameters. To fine-tune the pre-trained models, we replace the last fully-connected layer with two 1×1×1 convolutional layers (of 128 and 7 neurons) trained from scratch, with dropout before and in between. These layers operate on each frame independently and the spatial dimensions are already average-pooled to 1×1.

2.2 Experimental setup

2.2.1 Dataset

For most experiments of this study, we consider the visual part of the audio-video EmotiW challenge (Dhall *et al.*, 2018) as our evaluation task for spatiotemporal facial expression recognition. The EmotiW challenge has supported the evolution of the field since 2013 and provides rich literature. The underlying database is Acted Facial Expression in the Wild (AFEW), presented in Dhall *et al.* (2012) and illustrated in Figure 2.3. The task is to classify video samples by assigning each of them a single emotion label from the six universal emotions, Angry, Disgust, Fear, Happy, Sad & Surprise, as defined by Ekman & Friesen (2003), or Neutral expression category. As discussed in the deep FER survey of Li & Deng (2018), "the categorical model that describes emotions in terms of discrete basic emotions is still the most popular perspective for FER, due to its pioneering investigations along with the direct and intuitive definition of facial expressions".

The performance metric is the classification accuracy (video-level rank-1 accuracy). We do not consider the audio information in our study. Because of the competition aspect of the literature related to the AFEW dataset, it is difficult to find evaluations of models based only on the visual component of the dataset, without the audio information. We thus provide some baseline results for 3D networks on AFEW.

Figure 2.3 Examples of movie scenes from the emotion video dataset AFEW

Each data sample is composed of a video sequence labeled with one of the seven classes. The video samples were collected from movies, providing close to real world environment. The Test set also contains reality TV shows data. The AFEW video dataset was collected and labelled semi-automatically. Video proposals were extracted from the movies by looking for expression-related keywords in the movie subtitles. Human labelers filtered relevant proposals with notably "visible presence of subjects" and "display of meaningful expressions". Video were then annotated with a single class-label by a human labeler, based on the dominating expression in the video (with video, audio and subtitle information).

Due to the complexity of real-world facial expressions, and the difficulty to describe them with only seven emotion categories, the classification accuracy of AFEW by human observers has been measured around 60% (Kächele, Schels, Meudt, Palm & Schwenker, 2016; Vielzeuf, 2019). This shows the ambiguity and subjectivity of such a categorical framework for emotion recognition.

The dataset has been constructed in a subject independent manner. Actors and movies in the Test set do not appear in the Train and Validation data. The Train set counts 773 samples from 67 movies, with 228 actors. The Val set counts 383 samples from 33 movies, with 134 actors. Videos have resolution 720×576 pixels and 25 frames per second. AFEW video duration ranges from 0.6s to 5.4s (between 16 and 128 frames), with an average of 2.5s.

Created from movie samples, AFEW provides close to real-world data, with a wide range of challenges due to the variation in head poses and movements, illumination and backgrounds. The baseline for the sub-challenge is based on computing LBP-TOP (spatiotemporal) descriptor and using SVM for classification, achieving 38.81% accuracy on the validation set.

2.2.2 Evaluation protocol

In the EmotiW challenge, video-level classification accuracy on the test set is used as evaluation metric. As the dataset is published for the EmotiW challenge, we do not have access to the test data, but only Train and Val sets. We report results on the Validation set, and thus use a separate subset for validation. We split the Train set into training and validation subsets 80% and 20%, using the original Val set as our testing set, for performance evaluations. Data examples present in the testing set and in the training/validation set belongs to mutually exclusive movies and actors. This allows us to use a subject independent evaluation.

The folds for training and validation subsets are generated with 80/20 ratio for each category, preserving the original class distribution (the dataset is not balanced). We use the training subset to update the model parameters (computing loss for gradient descent). The validation subset allows us to tune hyperparameters. It is also important to use the same training sets across experiments in order to compare results. In order to provide a fair comparison of different models and architectures, we execute a grid search of learning rates hyperparameterization for each model. To reduce the effect of randomness, we reproduce experiments 9 times with different random seeds and training/validation folds, and report the standard error for all experiments. The splitting between training and validation sets can have a great influence on the performance. For instance, the distribution of subjects across the sets can vary and impact the model's generalization capacities.

2.2.3 Training details

2.2.3.1 Face extraction

The videos are provided as *.avi* files while our deep-learning framework requires image arrays. We first sample videos to series of frames keeping the original spatial and temporal resolutions: 720×576 pixels and 25 fps. Faces are then detected and aligned using the Seeta Engine from Liu, Kan, Wu, Shan & Chen (2017).

This process allows faces to have approximately the same size and pose, unlike the original database in which a face can occupy the whole frame or just a small area (depending on camera shot). This face-crop process operates at frame-level, without tracking. The frame is ignored if no face is detected. In case several faces are detected we only keep the largest one. The average bounding box of source crop has size 265×265, while the original videos have 720×576 pixels.

2.2.3.2 Data augmentation

We perform data augmentation with random rotation, resize crop, horizontal flip and color jitter[3]. The preprocessing is as follows in all experiments, unless noted otherwise: horizontal flip with 0.5 rate, random rotation between -20 and 20 degrees, random crop of the image with half spatial size, color jitter with brightness, contrast and saturation and hue (0.2, 0.2, 0.2, 0.1 settings respectively). This video preprocessing is consistent across all frames within a video sample. Frames are resized to 112×112 for the 3D model. We apply normalization at pixel level to mean 0 and standard deviation to match the model's pre-training.

2.2.3.3 Sample duration standardization

For the validation and inference processes, our fully-convolutional architecture with global pooling, working with arbitrary-length videos, is equivalent to using the 3D CNN on every

[3] https://github.com/hassony2/torch_videovision

possible windows of k consecutive frames in the video sample, where k is the model's temporal receptive field. The results can then be aggregated with temporal pooling. During training however, we still need uniform batches.

Literature generally presents as a difficult challenge the problem of modelling arbitrary-length videos with a fixed number of parameters (Ng *et al.*, 2015; Karpathy *et al.*, 2014). This is surprising, especially in the work of Karpathy *et al.* (2014) because they use spatiotemporal convolutions, which implies a common structure in space and time dimensions.

Possible solutions to obtain videos of equal duration, notably in order to work with input batches are:

– re-sampling the video, adapting the frame-rate to the video length,
– using contiguous clips of fixed duration extracted from the video,
– padding shorter videos to the size of the longer video (with black frames or a loop effect).

It is unclear why it is common to rescale images spatially and not frame sequences temporally. We can note that images have different scales by construction, from their capture, with distance and zoom affecting the apparent size of an object. When working with FER however, face alignment reduces this effect. On the temporal dimension, all videos are captured with the same frame rate, so the temporal dimension is homogeneous across all videos. This different treatment of time and space in data processing contrasts with the unified spatiotemporal feature extraction inherent to 3D CNNs.

Convolutions and max pooling allows for spatial invariant in CNNs. With the temporal dimension however, the duration is generally smaller than the spatial size, so pooling reduces the temporal resolution rapidly. As shown in Figure 2.4, the AFEW videos range from 0.6s to 5.4s. Also, the temporal dimension, being roughly 10 times shorter than the spatial dimension (if we consider an average of 64 frames and size 576 pixels), is much more sensitive to subsampling. As spatiotemporal models are usually thought as an adaptation of 2D models to the spatiotemporal domain, the architectures are biased toward attributing more importance to space than time. This is especially true with inflated 3D models, as discussed in Section 1.2.5.

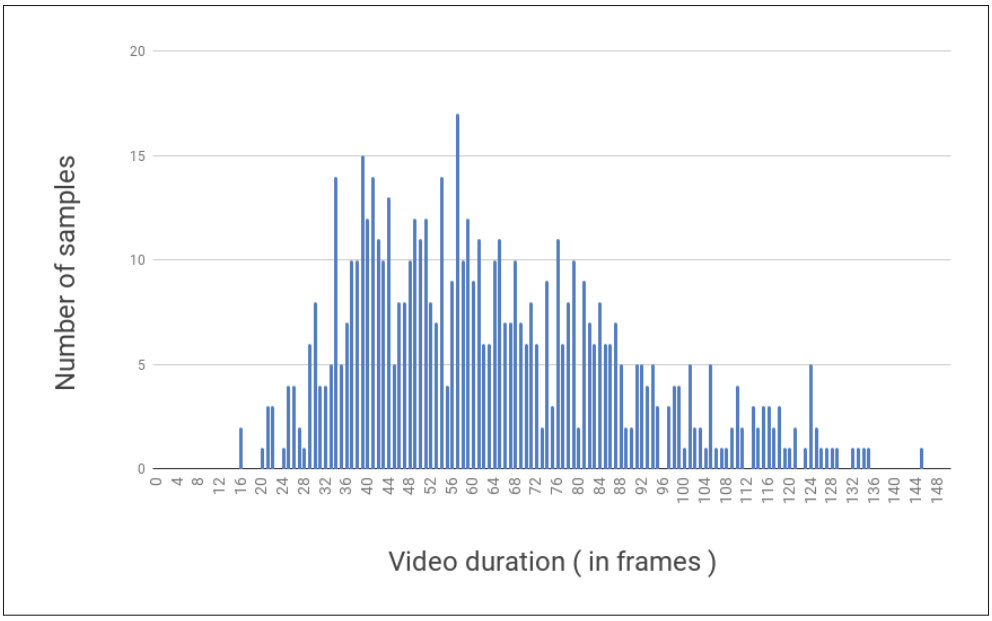

Figure 2.4 Distribution of sample duration in our face-cropped AFEW dataset (videos have 25 frames per second)

We compare two approaches to duration standardization during training:

– fixed-length clip extraction, preserving temporal resolution but leaving out entire parts of the original video;
– temporal resizing, better representing the entire video but making temporal resolution (thus speed) inconsistent across samples.

As discussed in Section 1.2.3, fixed-length clip extraction is the commonly used technique in literature for spatiotemporal deep learning. We also experiment the temporal resizing of videos. Frames are sub-sampled from the entire video, thus representing the scene in its entirety. The result is rich in spatial information, and the temporal dimension is accelerated or slowed, making the temporal pattern recognition more difficult. Also, while fixed-length clip extraction allows to sample different clips starting at randomly selected starting frames, resizing a given video sample to a fixed length always produces the same result.

We do not consider padding or looping videos to a large enough duration, which could preserve both the temporal resolution and the general aspect, because GPU memory requirements are limiting.

2.2.3.4 Training hyperparameters

We use ADAM optimizer with different learning rates for the pre-trained 3D model and the 3D classifier. For every experiments we set the weight decay to 0.3 and 0.2 for the 3D CNN and the classifier respectively. We use dropout with 30% drop rate on the input and 90% drop rate on features between the 3D CNN and the classifier to prevent overfitting. The validation subset allows us to choose the best hyperparameterization during grid search, and to select the limit epoch for training. We stop the training when no improvement on the validation loss is observed for more than 10 epochs.

2.3 Results and discussion

2.3.1 Feature and score pooling

We compare the spatiotemporal aggregation of clip features and clip predictions, with max and mean operators. The features representations, with 2048 or 512 channels (for ResNet and VGG), should describe the videos more precisely than the 7 class scores. However, the class scores are directly oriented toward our problem, while the feature extractor has been pretrained for action recognition.

Table 2.1 provide results with different pooling configurations with the 3D ResNet pretrained on action recognition, and our i3D VGG pretrained on face recognition. We note that max-pooling on feature representations performs very poorly for the ResNet model. This is not observed with the i3D model, and the reason is probably related to the gap between the pretraining and the final task. The max operation on the temporal dimension might produce very noisy gradients, by focusing learning on specific frames instead of the overall emotion of the video.

Table 2.1 Accuracy for max and average pooling at feature and score level with ResNet3D and i3D VGG on AFEW

Pooling type	**3D-ResNet Acc. (%)**	**i3D-VGG Acc. (%)**
3D max features	31.72 ±0.93	44.10 ±0.45
3D avg features	38.29 ±0.39	45.08 ±0.24
3D max scores	37.08 ±0.62	44.91 ±0.43
3D avg scores	36.72 ±0.48	43.82 ±0.24
2-step pooling	38.53 ±0.49	45.43 ±0.34

For both models, 3D average pooling on features is the best of the four configurations of unified spatiotemporal pooling. Having the classifier work on a global condensed representation of the entire video volume allows for better decisions, compared to deciding class scores only from local information in the spatiotemporal volume.

The performance of the 2-step pooling strategy shows that max pooling works well with the spatial dimension, while the temporal dimension benefits from the average strategy. This is coherent with what we observe in literature. The max-pooling layers are used throughout the architecture to reduce the spatial size, as with 2D CNNs. The models usually benefit from keeping the temporal dimension rich enough to obtain a sequence of frame-level scores that are aggregated at the end, with average pooling. Indeed, pooling the spatial dimension early (on feature-level and before) is necessary as it increases the receptive field of following convolutions. As the temporal dimension is smaller (16 to 64 frames), the hierarchy of convolutional kernels already has a sufficiently large receptive field.

We verify this idea by removing the temporal stride in the 3D ResNet model. The resulting model produces a dense temporal representation, with a classification score per input frame. We then apply temporal average pooling as before. Table 2.2 shows that the 3D ResNet model benefits from temporally dense convolutions, with more precise representation of the video dynamics. Note that our i3D model was already designed without temporal stride.

Table 2.2 Performance of a temporally dense 3D ResNet obtained by removing temporal stride in convolutions

Model	**Accuracy (%)**
Original (temporal stride 2)	38.53 ±0.78
Dense (temporal stride 0)	39.25 ±0.40

Even though our objective is to work with spatiotemporal 3D convolutions, unifying the processing of space and time dimensions, we note that they can also benefit from different treatment, because they have different properties in practice.

2.3.2 Temporal pooling temperature

To further study the impact of average and max pooling on the temporal dimension of the score predictions, we experiment with softmax pooling, as proposed by McFee *et al.* (2018). The adaptive softmax has a learning component that adapts the softmax temperature to the data, with a parameter β_c for each class c. The class score y_c for the input video is obtain by weighted pooling of temporal class scores $x_{t,c}$:

$$y_c = \sum_{t=1}^{T} \frac{\exp(\beta_c x_{t,c})}{\sum_{j=1}^{T} \exp(\beta_c x_{j,c})} x_{t,c}. \tag{2.1}$$

When $\beta_c = 0$, all the weights are equal to $\frac{1}{T}$, so the operator is equivalent to average pooling. When $\beta_c \to +\infty$ all the weight will be assigned to the highest scoring t, as in max pooling. Equation (2.1) is therefore a generalization of max and average pooling, parameterized by a factor β. The parameters are unconstrained and can go beyond the $[0, \infty[$ range that describes pooling behaviors in between the average and max paradigms. Pooling with a negative temperature could be relevant to enhance the low activations, but we do not observe this behavior in experiments.

The parameters are initialized to 1, as in the usual softmax function. After training, we obtain the following β parameters:

- Surprise: 3.022;
- Fear: 2.263;
- Angry: 2.052;
- Disgust: 1.542;
- Sad: 1.409;
- Happy: 0.562;
- Neutral: 0.479.

The Surprise class has the highest temperature, which means that the model associates this emotion with short events. On the contrary, the Neutral class has a low temperature because the model identifies it as a state more than an action. These results are as we expected after reading the related discussion from McFee *et al.* (2018).

However, this behavior does not reproduce easily and we usually fail to learn such interesting softmax parameters. Even then, we do not observe significant improvement in the classification accuracy. Adaptive softmax temperature within the training clip is very limited. We can imagine that the technique would perform better for pooling across clips, but this is not possible in our case as the training uses only one clip so these temperature parameters could not be optimized. This also raises the issue of the consistency of the pooling strategies during training, with short clips, and during testing, with long videos. Applying softmax pooling within 16-frame windows or on 250-frame videos produces very different distributions. This issue motivates the study in Chapter 3.

2.3.3 Video duration standardization

Table 2.3 presents results obtained with different methods producing training videos of equal duration for building batches of homogenized samples. Both methods are effective but results are significantly better with the clip-extraction method. Training from short clips prevents

long-range dependencies but allows for detecting precise local patterns, thanks to the unaltered frame rate.

This explains why temporal clipping is the commonly used training strategy, as described in Section 1.2.3. The iterative learning from random clips containing only part of the video benefits from the higher, unaltered and homogeneous temporal resolution. Also, the randomness of clip extraction provides an additional form of data augmentation.

Table 2.3 Comparison of duration standardization methods for building batches of training samples

Standardization	**Accuracy (%)**	**Epochs**
Clip extraction	46.53 ±0.36	35.2 ±5.3
Temporal resize	45.07 ±0.50	35.6 ±4.2

2.3.4 Clip length

Table 2.4 Accuracy for different lengths of training clips with 3D-ResNet and i3D-VGG models

Clip length (frames)	**3D-ResNet Acc. (%)**	**i3D-VGG Acc. (%)**
8	35.72 ±0.96	44.91 ±0.75
16	40.56 ±1.17	46.53 ±0.62
32	36.29 ±0.41	47.10 ±0.71
64	28.20 ±1.19	-

We compare different duration of training clips. Shorter clips allow to focus on more elementary patterns, which should help for generalization. Longer clips reduce the clip-level labelling noise, and thus noise in training gradients. More importantly, training with more frames allows to learn longer-range temporal patterns, in the limit of the model's receptive field. The receptive field size of the original models is 37 frames (with jump 16). When we remove the stride of all layers, the model performs convolutions densely, providing more outputs along the temporal axis, and the receptive field covers 15 frames.

We observe from Table 2.4 that the 3D ResNet's performance decreases with longer training clips. Our temporally dense model, with reduced temporal receptive field cannot take advantage of long training clips for long-range patterns, but should benefit from the reduced labelling noise. As the model was pretrained with 16-frame clips, it is biased towards feature-extraction in this range. Also, increasing the clip duration necessitates to decrease batch size, because of GPU memory limitations, which has a negative impact because of batch normalization layers Ioffe & Szegedy (2015).

On the contrary, the i3D model isn't biased by pretraining, and also has temporally dense convolutions, but with a wider receptive field. The VGG-16 architecture contains several spatial max-pooling layers. When inflating, these pooling layers are kept spatial and we do not use temporal pooling within the feature extractor. The global temporal receptive field is 27 frames. The model should thus be able to benefit from training clips of 32 frames. However, 32 frames might not be enough to significantly reduce the labelling noise.

This is confirmed by results reported in Table 2.4. However, increasing the clip size increases the training time in two ways. First, more frames have to be processed, increasing computation time proportionally. Second, as we have to reduce batch size for memory constraints, we reduce parallelization, which requires increasing the number of epochs. Thus we do not explore further the increase of clip size. For practical reasons, we consider 16-frame clips provide a good compromise to produce more experiments.

2.3.5 3D inflation

We evaluate the inflated model trained from scratch on AFEW and the model pretrained on VGG-Face + FER2013, and fine tuned on AFEW. With a resulting classification accuracy of 24.28%, our experiments show that the model can not be trained from scratch on AFEW. The 2D "warm-start" has a key role in allowing the model to learn discriminant spatiotemporal patterns.

2.3.6 Discussion

Despite our intensive use of data augmentation and dropout, we observe that the ResNext-101 architecture overfits on our small dataset, and reaches training accuracies above 70%. The ResNet-18 achieves better performances with lower training accuracies, indicating better generalisation.

Because of audio-video nature of the EmotiW challenge, literature doesn't provide a lot of description and results for video only. Also, challenge submissions regroup several networks for image and audio, and details are usually not available for the performance of individual components. Table 2.5 regroups results from 3D CNNs on the validation set of the AFEW dataset. Our models are in range of these results.

Fan *et al.* (2016) and Ouyang, Kawaai, Goh, Shen, Ding, Ming & Huang (2017) use the small C3D model of Tran *et al.* (2015), pretrained on sport videos. Vielzeuf *et al.* (2017) managed to improve performances with a recurrent neural network to add another level of temporal analysis, and with their weighted C3D which is a type of attention used for the training process. Also we should note they use an industrial proprietary face alignment method. The 3D VGG network of Lu *et al.* (2018) is heavier to train and performs relatively weakly. Our experiments show that leveraging pretraining from 2D face and emotion datasets enables this 3D model to obtain good accuracy.

Table 2.5 Related results from the literature for EmotiW challenge on the AFEW dataset

Reference paper	**Model type**	**Acc. (%)**
Fan *et al.* (2016)	C3D sport1m from Tran *et al.* (2015)	39.7
Ouyang *et al.* (2017)	C3D sport1m from Tran *et al.* (2015)	35.2
Vielzeuf *et al.* (2017)	C3D on central window	38.7
Vielzeuf *et al.* (2017)	C3D on random window	34.0
Vielzeuf *et al.* (2017)	LSTM C3D (no overlap)	43.2
Vielzeuf *et al.* (2017)	Weighted C3D (no overlap)	42.1
Lu *et al.* (2018)	3D VGG-16	39.4

CHAPTER 3

TEMPORAL STOCHASTIC SOFTMAX FOR 3D-CNN TRAINING

The work presented in this chapter has been the focus of our publication at the IEEE Winter Conference on Applications of Computer Vision, WACV 2021, Ayral, Pedersoli, Bacon & Granger, 2021, *Temporal Stochastic Softmax for 3D CNNs: An Application in Facial Expression Recognition*,

Deep Learning (DL) models have been successfully applied in many visual recognition tasks, including detection, classification, tracking, and segmentation, and currently achieve state-of-the-art performance on several image-based benchmarks (Karpathy *et al.*, 2014; Carreira & Zisserman, 2017). Spatiotemporal recognition, in which appearance and motion features play a complementary role, remains a challenging problem in real-world applications. While many DL models are based on spatial feature extraction, specialized mechanisms are needed to manage spatiotemporal data.

In facial expression recognition (FER), the output produced by a 2D Convolutional Neural Network (CNN), e.g. VGG or ResNet models, in response to a sequence of frames is typically aggregated or processed by a recurrent neural network which, as seen in AVEC and EmotiW competitions (Dhall, 2019), can provide high-level performance (Li *et al.*, 2019; Liu *et al.*, 2018a; Lu *et al.*, 2018). In contrast, 3D CNNs can process a clip as a single input, and jointly analyze appearance and motion to encode spatiotemporal relationships (De Melo, Granger & Abdenour, 2020a; De Melo, Granger & López, 2020b; Praveen, Granger & Cardinal, 2019). 3D CNNs have also been integrated as components in FER systems, not always performing well on their own with limited data (Fan *et al.*, 2016; Vielzeuf *et al.*, 2017; Lu *et al.*, 2018). However, recent studies have shown the relevance of 3D CNNs for video recognition, highlighting the importance of adopting appropriate training strategies, and integrating extra training data through transfer learning (Carreira & Zisserman, 2017; Hara *et al.*, 2018; Xie, Sun, Huang, Tu & Murphy, 2018). Along these lines, our work is focused on efficient training strategies for 3D CNNs.

The computational requirements of 3D CNNs represent an important challenge in video recognition applications. Motion adds an extra dimension to model representations (i.e., inputs and feature tensors are much larger), and significantly increases the computational and GPU-memory requirements for training a DL model. To address this issue, state-of-the-art 3D models (Carreira & Zisserman, 2017; Hara *et al.*, 2018) are trained with short, randomly sampled training clips, which is a stochastic approximation of temporal average pooling (see Section 3.2.1). At inference time, as memory requirements are reduced, temporal average pooling is used. Training with short clips has the advantage of mitigating issues related to GPU memory and video length. The efficiency of this technique in practice suggests that modeling long-range temporal dependencies is not needed in most cases to achieve accurate spatiotemporal recognition. However, a uniform selection of training clips from real-world videos, enforcing equal importance to all frames, raises other issues for training and inference. Clips extracted from a video captured "in the wild" are not all equally relevant, because of inherent characteristics of the tasks or noise in capture conditions. Assigning a global video label to short clips generates noise, and some clips can even be misleading because they do not represent the general aspect of the video. This has important implications for both training and testing phases. For instance, in FER applications, expressions captured in videos vary significantly depending on subjects and capture conditions (e.g. illumination and pose). As a result, parts of a video may not contain any relevant information. Also, the expression intensity in FER videos varies through different states, typically onset, apex and offset (Zhao, Liang, Liu, Li, Han, Vasconcelos & Yan, 2016; Kamarol, Jaward, Kälviäinen, Parkkinen & Parthiban, 2017), and not all these states provide the same discriminative power for spatiotemporal recognition. Moreover, most of the video is typically dominated by neutral state and does not correspond to the sequence-level expression label. To avoid investing computational resources on training with uninformative clips, and to reduce the performance limitation incurred by training on incorrectly labeled clips, it is preferable to learn to sample the most relevant clips.

Contribution: We present a new stochastic softmax method to efficiently train 3D CNNs for spatiotemporal recognition, with videos of arbitrary length and sequence-level labels. This

method leverages a stochastic approximation of softmax temporal pooling for efficient learning with sampling of relevant clips. Softmax sampling weights are estimated iteratively during training, with lower variance than the REINFORCE method (Williams & Peng, 1990), thereby leading to better results. Although uniform clip sampling is often used for its simplicity, empirical results on several FER datasets show that the proposed temporal stochastic softmax provides a better training approach for 3D CNNs, achieving a higher level of accuracy, and a shorter training time.

3.1 Temporal weighting

Although the practical analysis of Huang *et al.* (2018) shows that 3D CNNs mostly perform spatial feature extraction, relying heavily on appearance and neglecting temporal patterns, it also suggests that selecting interesting frames produce better performance than classification from the entire video. The authors hypothesize that long-term patterns are not crucial for action recognition, but a wide receptive field is important to select the most relevant frames. Our work is in line with these challenges, as our proposed softmax training consists in an efficient weighting of short video clips for 3D CNNs, with sampling during training and through pooling at test time.

Works related to ours are ones that study techniques to improve inference through a weighted aggregation method, or to improve training with better sample, clip or frame selection. Temporal aggregation of features for 3D CNNs is usually performed with average pooling (Carreira & Zisserman, 2017; Hara *et al.*, 2018; Diba, Fayyaz, Sharma, H., Arzani, Yousefzadeh & Van Gool, 2018; Varol *et al.*, 2018), as it can easily be used with short-clip training (more details are provided in Section 3.2.1).

The importance of the pooling strategy has been discussed in Section 1.2.4.3. To help find the most adapted type of pooling, softmax is proposed as a parameterizable generalisation of max and average pooling. The adaptive softmax pooling of McFee *et al.* (2018) is able to optimize the temperature parameter for each class, with a data-driven approach. This allows to tune the

pooling strategy to the task dataset and answers the issue of discrepancies in event duration of different classes.

The pooling strategy used for temporal feature aggregation also has a great influence on training, by deciding which part of the input will generate gradients. For 3D CNNs, as training is performed on short clips (usually 8 to 64 frames), the receptive field of the aggregation mechanism is limited, and the pooled features have a different distribution than at test time, with longer videos. Consequently, other weighting methods have to be developed.

Studies on importance sampling (Katharopoulos & Fleuret, 2018; Wang, Song, Leung, Rosenberg, Wang, Philbin, Chen & Wu, 2014) made clear that all samples do not have the same relevance to the training. They design methods to build training batches with samples that maximize learning, measured as the norm of weight update. The objective is to avoid wasting time on samples that are already well managed by the network, and focus on difficult samples, reducing the variance of the stochastic gradients during training. In Wang *et al.* (2014), buffers are used to store and sample relevant training images. A relevance score determines the probability of selecting each image to construct inputs for the triplet loss, because it would be "computationally prohibitive and sub-optimal to use all the triplets" (Wang *et al.*, 2014).

However, for most video classification datasets, e.g. AFEW (Dhall *et al.*, 2012) and Kinetics (Kay *et al.*, 2017), labels are not available for each of the frames or clips, but only at the video level. Related issues are discussed by Zhu, Hu, Sun, Cao & Qiao (2016), as they consider short-clip training as a weakly supervised learning problem within the action recognition task with video-level labels. During training, they provide the model with the ability to learn only from the most informative clip from a spatially and temporally sub-sampled set of the original training video. In this multiple instance learning (MIL) framework, a bag of several clips is fed to the model and temporal max pooling is used to learn only from the best-scored clip. This is a way of providing better training data by selecting the most informative temporal windows *a posteriori*. This approach still requires the use of several clips per sample at each epoch, which is problematic for 3D models. To be able to select clips before evaluating them with the entire

model, this method is complemented with a motion metric, computed offline. In the context of action recognition, the hypothesis is that relevant clips are the ones with more motion. This does not hold for FER (as very expressive faces can be static), so our method is purely based on classification scores, and we compute them online with a single training-clip per sample for each epoch, to save computation with the 3D CNN.

Also to cope with capture, trimming and labelling noise, the weighted C3D (Vielzeuf *et al.*, 2017) integrates a softmax layer to give more importance to relevant clips during training. All windows are evaluated in the early epochs of training and their scores are used to weight the training loss, reducing the effect of uninformative or wrongly labeled clips on the model parameter updates. The weighting strategy is amplified throughout training, from average to max. This principle of weighted training constitutes a basis of our work (although the weighting method is not very detailed in Vielzeuf *et al.*, 2017), with the idea of identifying relevant training clips for 3D CNNs. Yet we replace the loss weighting by sampling to avoid computing gradients that will be inhibited by softmax, and we change the distribution estimation method to remove the computational overhead of evaluation. This allows us to train a model with more parameters than the small C3D.

Another interesting work is the SCSampler (Korbar *et al.*, 2019), trained to select the best clips to feed to the classifier at test time. This external, light-weight, sampler is trained to predict the relative saliency of clips. By running the classifier on a few sampled clips instead of the entire video, the computational cost at inference time is reduced and classification accuracy is improved. This experimental study shows that classification score is a good metric of the informativeness of a clip. Indeed, the *sampler network* is trained to imitate the *oracle sampler*, which ranks clips according to the score they obtain under the classifier for the target class. As the authors explain, “in real scenarios the oracle cannot be constructed as it requires knowing the true label and it involves dense application of f [the costly action classifier] over the entire video, which defeats the purpose of the sampler” (Korbar *et al.*, 2019). We actually construct this oracle sampler and use it for training because labels are available, and we propose a way to avoid dense application of the classifier through iterative sampling (Section 3.2.3).

3.2 Uniform and softmax sampling

The main objective of this work is to perform a pooling operation that can select the most important frames of a video sequence. As presented in Section 3.1, softmax pooling seems to be the right candidate because it assigns a specific weight to each short clip of the video. However, standard softmax pooling requires an evaluation of all short clips of a video sequence. This is unfeasible for large 3D models as they require a large amount of memory and computation. In this section we show how to approximate softmax pooling during training such that it requires the evaluation of only one short clip per video at each training iteration. This makes the training much lighter, while optimizing the same objective function in expectation.

Section 3.2.1 presents the motivations and challenges of integrating softmax pooling into the usual short-clip training framework. Then, Section 3.2.2 discusses solutions for estimating temporal probability distributions iteratively from short clips. Finally, Section 3.2.3 provides the details of stochastic softmax, combining both training-clip sampling and softmax pooling, with a unified temperature parameter.

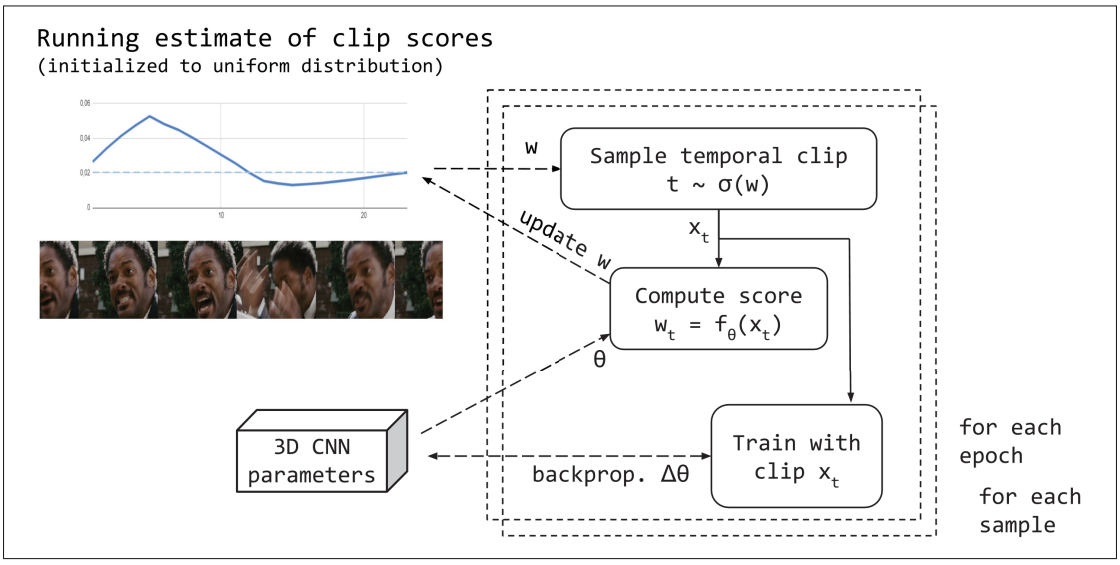

Figure 3.1 Illustration of weighted clip sampling for temporal stochastic softmax training. Within each video, training clips are sampled based on the softmax of their classification-score distribution. Running estimates of clip scores are updated at every iteration with the classification score of the selected clip, © 2021 IEEE

3.2.1 Softmax pooling with clip sampling

Our objective is to minimize the loss $\mathcal{L}$ of a parameterized classifier f on the training dataset. For each video x with label y, we minimize $\mathcal{L}(f(x), y)$. For video classification, a typical example of such loss is cross-entropy. If using temporal average pooling, the loss can be written as $\mathcal{L}(\frac{1}{T}\sum_{t=1}^{T} f_t(x), y)$ in which $f_t(x)$ represents the learned features associated to temporal position t in a video of duration T. Assuming that the temporal receptive field of the network is limited or that the features extracted for time t only depend on a small temporal neighbourhood (a clip), we can compute the loss as $\mathcal{L}(\frac{1}{T}\sum_{t=1}^{T} f(x_t), y)$ where x_t is a clip associated to time t. This assumption is implicitly or explicitly used in most of the recent work on 3D CNNs for video classification (Carreira & Zisserman, 2017; Hara *et al.*, 2018; Diba *et al.*, 2018; Varol *et al.*, 2018) because it enables the use of an approximation of the loss:

$$\mathcal{L}(\frac{1}{T}\sum_{j=1}^{T} f(x_j), y) \approx \mathcal{L}(f(x_t), y), \quad t \sim \mathcal{U}(1, T). \tag{3.1}$$

The loss is computed by sampling different clips throughout training. For each iteration, the loss of a video is approximated using a single clip that is uniformly sampled from each video. This produces significant reduction in computational complexity of each training step and in GPU memory requirements necessary to make the training of 3D CNNs possible.

Here we show that this sampling technique with cross-entropy loss is an upper bound of the real loss. Indeed, cross-entropy loss is convex, and based on Jensen's inequality, the average loss computed on all clips $\frac{1}{T}\sum_{t=1}^{T} \mathcal{L}(f(x_t), y)$ is an upper bound of the cross-entropy of the average pooled features $\mathcal{L}(\frac{1}{T}\sum_{t=1}^{T} f(x_t), y)$:

$$\frac{1}{T}\sum_{t=1}^{T} \mathcal{L}(f(x_t), y) \geq \mathcal{L}(\frac{1}{T}\sum_{t=1}^{T} f(x_t), y) = \mathcal{L}(f(x), y) \tag{3.2}$$

The average of clip-level losses is an empirical estimation of the expected loss $\mathbb{E}[\mathcal{L}(f(x_t), y)]$. Instead of computing the entire sum, the expected loss is approximated with a single sample t,

uniformly sampled as in Eq. (3.1). Thus, in expectation, the same loss is optimized during training. At test time, memory is less problematic as gradients are not computed, so inference is performed with the average of all video clips, thus a temporal average pooling.

Although this form of training with uniformly sampled clips can be effective, it restricts the temporal pooling to the average pooling strategy. Our work addresses this issue by proposing the more general softmax strategy in the training-clip sampling framework. Temporal average pooling assumes that each sub-region (clip for videos) of the pooled features contains information that is important for the task. It is expected to perform well when videos are short and entail exactly the category that we want to classify (Boureau *et al.*, 2010; McFee *et al.*, 2018). On the other hand, when a video is longer, complex and with possible noise, max pooling is expected to work better. In general, the optimal level of importance for the different parts of a video is unknown, as it depends on the task and the actual data. In this paper, we propose to use weighted pooling in which a weight p_t is associated with each temporal position t of the video. This resembles an attention mechanism (Xu *et al.*, 2015; Girdhar, Carreira, Doersch & Zisserman, 2019; Meng, Peng, Wang & Qiao, 2019; Aminbeidokhti *et al.*, 2019; Li *et al.*, 2018a), but instead of estimating attention weights with a learned layer, p_t is computed as a temporal softmax of the score associated to a given clip. The relative importance attributed to each clip depends on its classification score:

$$f(x) = \sum_{t=1}^{T} \frac{\exp(\gamma f(x_t))}{\sum_{j=1}^{T} \exp(\gamma f(x_j))} f(x_t) = \sum_{t=1}^{T} p_t f(x_t). \tag{3.3}$$

The softmax operator is parameterized by γ, the inverse temperature. When $\gamma = 0$, the weight vector $p = (p_1, p_2, ...p_T)$ is equal to the center of the unit simplex, with $p_t = \frac{1}{T}\ \forall t$, so the operator is equivalent to average pooling. In contrast, when $\gamma \to +\infty$ all the weight will be assigned to the highest scoring t, as in max pooling. Equation (3.3) is therefore a generalization of max and average pooling, parameterized by a factor γ. As with uniform clip sampling, we

provide an upper bound on the training loss obtained with softmax pooling:

$$p_t \sum_{t=1}^{T} \mathcal{L}(f(x_t), y) \geq \mathcal{L}(p_t \sum_{t=1}^{T} f(x_t), y) = \mathcal{L}(f(x), y). \tag{3.4}$$

The weighting factor p_t of softmax pooling, from Eq. (3.3), becomes a sampling probability distribution in stochastic softmax training. Indeed, for each video sample x, instead of weighting losses obtained from clips sampled uniformly, we directly weight the clip-sampling distributions. This way we select clips that are more important for training than with uniform sampling.

3.2.2 Estimation of softmax sampling distributions

In Eq. (3.3), we see that in order to compute p_t we need to evaluate $f(x_t)$ on all clips t of a video, which is computationally expensive and memory demanding. This is exactly what we want to avoid with a stochastic sampling strategy. Thus, instead of computing p as in Eq. (3.3), we introduce a new variable $q = (q_1, q_2, ..., q_T)$ that estimates p for each video x. During training, for each video, q is defined as minimizing the loss. A straightforward way to estimate q is by using REINFORCE (Williams & Peng, 1990). We consider the loss as an expectation over time, sampled with q. Thus, its gradient will be:

$$\begin{aligned} \nabla_q \mathop{\mathbb{E}}_{t\sim q}[\mathcal{L}(f(x_t), y)] &= \nabla_q \sum_{t=1}^{T} \mathcal{L}(f(x_t), y) q_t \\ &= \mathop{\mathbb{E}}_{t\sim q}[\mathcal{L}(f(x_t), y) \nabla_q log(q_t)]. \end{aligned} \tag{3.5}$$

For each training video sample, we create a parameter vector, initialized to a uniform density, that defines the sampling probability of each clip.

Unfortunately, the gradients estimated with REINFORCE have high variance and, even when using a baseline of the expected cumulative reward, the updates of q_t are too noisy. The potential

benefits of sampling are therefore lost by a poor estimation of q. Results and limitations of REINFORCE for the estimation of sampling distributions are discussed in Section 3.3.2.

As the distribution parameters are essentially pushed to favour the selection of high scoring (low loss) training clips, it is possible to "shortcut" the REINFORCE optimization building the sampling distributions directly from the clip scores. Thus, we propose to estimate q directly in a close form, without the use of gradients. Since p_t is calculated as softmax of $\gamma f(x_t)$, it is possible to store the values of $w_{x,t} = f(x_t)$ directly, and apply softmax when an estimation of q is needed to select the training clip. The probability distribution q is therefore computed from running estimates of scores evaluated at different iterations. This approach is quite simple, does not rely on a noisy estimation of the gradients, and works well in practice.

The study of softmax presented in Gao & Pavel (2017) shows that the softmax function σ constitutes the mixed strategy with maximum entropy that maximizes payoff. Considering a strategy x and scores z,

$$\underset{x}{\operatorname{argmax}}[x^\intercal z - \lambda^{-1} \sum_j x_j \log(x_j)] = \sigma(z). \tag{3.6}$$

During weighted training, softmax sampling maximizes the classification score for the correct label (providing relevant clips to the model), while keeping diversity in clip sampling to avoid overfitting (Gao & Pavel, 2017). The logit equilibrium is obtained when the scores and the payoffs converge to a fixed point ($U(\sigma(z)) = z$, with a score variable z, a strategy $\sigma(z)$ and a payoff function U). Building temporal distributions from training scores, can be seen as plugging in the logit equilibrium, to shortcut a slow and high variance learning rule.

3.2.3 Implementation of stochastic softmax

Videos are classified by convolving the 3D CNN in the temporal dimension, over all possible overlapping windows, and implementing a temporal pooling mechanism on the resulting clip-level scores (Carreira & Zisserman, 2017). We combine this evaluation scheme with single-clip extraction during training, as only one clip is used from each video at every epoch.

3.2.3.1 Weighted clip sampling

The sampler S, extracts a clip of F contiguous frames at temporal position t from a video x of arbitrary length L. The sampling mechanism can be formulated as:

$S : R^{L\times 3\times H\times W} \mapsto R^{F\times 3\times H\times W}$, with $F \leqslant L$, and $H \times W$ is the spatial resolution of the data and we consider three colour channels. At every epoch of training, we construct batches of training clips. One clip is sampled from each training video. There are $N = L - F + 1$ possible clips to extract from a given training example. Videos are padded to contain at least F frames.

With uniform sampling we have:

$$p(S(x) = x_t) = \frac{1}{L - F + 1} \; \forall x_t \subset x. \tag{3.7}$$

With weighted training, the temporal sampling probability distribution of each video is computed from its classification scores. In the context of a deep-learning classifier, we consider that inference class scores represent a good measure of a clip's informativeness and relevance to the task (Zhu *et al.*, 2016; Vielzeuf *et al.*, 2017). Specifically, for a given training clip, we use the score corresponding to the target label. In this sense, our method is similar to using the Oracle Sampler conceptualized in Korbar *et al.* (2019) at training time. This strategy minimizes the training loss by selecting the best scoring clips. The aim is to also improve the validation accuracy and reduce training time by learning from informative clips, without irrelevant and noisy frames.

Let w_x be the temporal sequence of $N = L - F + 1$ classification-score estimates corresponding to the temporal responses of the classifier convolved over x. Then $w_{x,t}$ is the estimated classification score for training clip x_t. This score will be the base of our clip weighting. Temporal softmax

sampling follows the formula:

$$p(S(x) = x_t) = \frac{\exp(\gamma w_{x,t})}{\sum_{n=1}^{N} \exp(\gamma w_{x,n})} \; \forall x_t \subset x. \tag{3.8}$$

The principle of stochastic softmax training is summarized in Eq. (3.8). At test time, the relative importance attributed to each clip through temporal pooling is defined as the weighting factor computed from the softmax function in Eq. (3.3). During training, the temporal weighting is implemented as a sampling probability which translates into a frequency of occurrence in the training iterations. Instead of classifying entire videos and applying temporal weights to the loss afterward, computation is exclusively focused on the selected clip.

3.2.3.2 Live update of distributions

A naive and accurate approach to build probability distributions would be to score every possible training clips for each video sample at every epoch. Running inference on the entire training videos at every epochs would be very expensive. Instead, the proposed approach stores running estimates w of the score distribution of each training video, and updates them around the corresponding temporal location at every epoch, as video clips are sampled. Through iterative clip sampling, we obtain information on the temporal classification score distribution of each video and use it to build its temporal clip selection probability distribution. An overview of the training process is presented in Algorithm 3.1.

At every epoch, a single clip is selected from each video in the training dataset. We apply inference on the sampled clip without data augmentation (as this would introduce noise in the score distribution), and separately use a copy with data augmentation to train the model. The importance of evaluating scores from "clean" clips is further discussed in Section 3.3.2.

Algorithm 3.1 Stochastic softmax training, © 2021 IEEE

Initialize w with uniform distributions;
foreach *epoch* **do**
  foreach *video x in training set* **do**
    compute the clip sampling distribution from w_x with Eq. (3.8) ;
    sample clip x_t, with $t = S(w_x)$;
    compute the score for the correct class y: $w_{x,t} = f(x_t)[y]$;
    update w_x around sampled location t ;
    train with clip x_t by back-propagation ;
  end foreach
end foreach

3.2.4 Training phases

Efficient training-clip sampling is highly dependant on the accuracy of the temporal distributions. Since the estimate w is built iteratively and the model is trained simultaneously, it could take a long time for clip sampling to become interesting. Waiting for all clips to be evaluated before starting to sample them efficiently would require an unreasonable number of epochs. Also, we do not want to update distributions with classification scores obtained from an untrained model. Therefore, we implemented several mechanisms to bootstrap the distributions, to make them representative of the informativeness of clips as early as possible during training, without introducing heavy computational overhead. We decompose the training process into three simple steps relative to the sampling mechanism: first, *warm-up* with uniform sampling and no distribution updates, then, *exploration* with deterministic sampling and initialization of distributions, and finally *exploitation* with softmax sampling and distribution updates (as described in Section 3.2.3). The number of epochs associated to each of these phases can be adapted to the task, based on the total duration of training, the average video length and the clip size.

• **Classifier warm-up:** During the first 3 epochs, uniform sampling is used without updating the distributions. This warm-up time allows the model to jump from 14% to 25% validation accuracy

on AFEW, which is about half of the final accuracy. After the 3rd epoch, the classification scores are far more reliable to update sampling distributions.

• **Deterministic exploration:** Before starting to exploit the sampling mechanism, we need to build entire temporal distributions for each video in order to have information on the relative importance of clips. One solution is to compute classification scores on the entire training videos with an inference step. This would be very expensive as a single training video can have hundreds of frames, and the number of possible F-frame overlapping clips to evaluate is $L - F + 1$. Also, the model's temporal receptive field and the duration of training clips are not equal, so using score vectors computed convolutionally from long videos might not be reliable and wouldn't be coherent with the scores obtained from short clips in the exploitation step. Instead, we propose a lighter exploration step that integrates smoothly into our framework. For 5 epochs, we sample training clips deterministically from uniformly spaced temporal locations. The selected clips are used for training and provide classification scores to initialize the corresponding $\frac{1}{5}$ of the distributions. In our implementation, the score is obtained before the back-propagation. Evaluating the clip right after training on it would bias the distributions toward rewarding most fitted clips, while we aim at evaluating their informativeness. Overall, this exploration step enforces a diversity of clips in the early stages, which is important for building representative distributions for efficient clip-sampling in the rest of training.

• **Weighted sampling – exploitation:** Then, for the main part of the training process, training clips are sampled stochastically, based on the softmax probabilities computed from w. We keep updating the distributions throughout training. At every epoch, a single clip is selected for each video in the training dataset. The clip is used for training (forward pass and back-propagation) and also to refine the sampling distribution based on the current state of the model.

We employ a propagation mechanism to update several clip probabilities from a single clip evaluation. As clips share a lot of frames with their neighbours, we update the distribution around the sampled temporal location. We consider this particularly important in our experiments because the number of clips in videos is generally greater than the number of training epochs.

We use linear interpolation centered on the selected clip and propagate to $F = 16$ frames on each side, with decreasing update weight further from the center. After training with clip x_t, obtaining classification score $f(x_t)$, we update the score w_{t+i} for all i in $[-F; F]$:

$$w_{x,t+i} = w_{x,t+i} + \frac{F - |i|}{F}(f(x_t) - w_{x,t+i}). \tag{3.9}$$

At the temporal location t of the sample, the update weight is 1, meaning the score is replaced with the new one. F frames further, the update weight is 0 and the distribution is unchanged.

The settings for the number of steps in each training phase and the method for smoothing distributions can be optimized and adapted to the task at hand, and require more attention in future work.

3.3 Results and discussion

3.3.1 Experimental methodology

We perform an ablation study of stochastic softmax training, and validate on several benchmarks of emotion recogntion, pain detection and action recognition, to evaluate the generalization power of the training method.

3.3.1.1 Datasets

• **AFEW:** The Acted Facial Expressions in the Wild dataset of emotion recognition (Dhall *et al.*, 2012) was used to evaluate the proposed and reference training methods. The task is to classify video samples by assigning each of them a single emotion label from the six universal emotions as described in Section 2.2.1. Created from movie samples, AFEW provides close to real-world data, with a wide range of challenges due to the variation in head poses and movements, illumination and backgrounds. Additional noise comes from camera motion, sometimes causing occlusion. Some of these issues are well addressed by the face alignment

process. Others can be managed with the proposed temporal weighting mechanism. The dataset was created with a semi-automatic extraction process, producing an imprecise trimming of movie samples. Videos are not always centered on the relevant emotional frames, different scenes can be present in a video, and multiple subjects can be present in the same frame.

• **UNBC-McMaster:** The UNBC-McMaster Shoulder Pain database (Lucey *et al.*, 2011) contains 200 image sequences capturing the spontaneous pain expressions of 25 subjects. Sequences vary in length from 48 to more than 500 frames. We follow Wu, Wang & Ji (2015) for the evaluation task. The dataset is used in a binary classification setup based on the Observed Pain Intensity (OPI) expert annotations. The 92 sequences with OPI = 0 constitute the negative samples (*No Pain*), while the *Pain* class is composed of the 57 sequences with OPI $\geq$ 3. Because of the class imbalance, the evaluation metric used for this task is the classification accuracy at Equal Error Rate on the Receiver Operating Characteristic curve (ROC-EER). We perform leave-one-subject-out cross-validation for the 25 subjects.

• **BioVid:** BioVid Heat Pain Database (Werner *et al.*, 2013), Part A, is a relatively large heat-pain detection dataset, with 20 frontal video recordings per stimulus level, for each of 87 subjects. Participants received four levels of painful stimuli (*PA1* to *PA4*), adapted to their subject-specific sensitivity. Videos with no stimulus are also present to constitue the *BL1* class (*No Pain*). Biomedical signals are also available but not used in our study. As opposed to UNBC-McMaster, BioVid provides objective labels based on the temperature of the heat-pain inducing device (with subject-specific levels). The task is thus much more difficult, as the objective is not to classify the directly observable expression but its source. All subjects do not react to pain with the same intensity, even though the stimuli are calibrated for each participants. The creators of the database studied this phenomenon and identified participants that did not react visibly to the pain-inducing stimulus (Werner, Al-Hamadi & Walter, 2017b). The BioVid videos are even more controlled than UNBC-McMaster and contain less head-pose variations and occlusion (Werner, Al-Hamadi, Limbrecht-Ecklundt, Walter & Traue, 2018). However, the number of irrelevant frames remains an issue, because videos are not trimmed to capture the specific expression but span a fix time window of 5.5 seconds based on the timing of the stimulus. Thus, results on

BioVid should specifically evaluate the ability of training-clip selection to favour expressive windows over relatively neutral states usually present at the beginning and end of every video. On this dataset, we use two different evaluation protocols from the literature. The first task is binary classification of Neutral (*BL1*) and highest level of pain (*PA4*), with 1740 videos per class. We perform 8-fold subject-independent cross-validation and report classification accuracy. The second setup is similar but the Pain class is extended to *PA3* and *PA4* videos (highest intensities), as proposed by Yang, Tong, López, Boutellaa, Peng, Feng & Hadid (2016). To work with the class imbalance (1740 *No Pain* and 3480 *Pain* videos), Area Under the Receiver Operating Characteristic Curve (ROC-AUC) is used as metric.

• **HMDB-51:** We evaluate the performance of our training method on an action recognition benchmark with the HMDB-51 dataset (Kuehne, Jhuang, Garrote, Poggio & Serre, 2011). It contains about 6800 videos labelled with one of 51 action categories. Each class contains at least 101 video examples. Most videos have a length between 45 and 120 frames, with an average of 94 frames. The dataset defines three splits of training, validation and test data. Results are averaged across the 3 splits of the dataset.

For all datasets, softmax training only requires sequences of RGB frames and the corresponding sequence-level class labels. Facial expression recognition requires an additional pre-processing step consisting of detection and alignment of faces in each video frame. We employed the SeetaFace Engine from Wu, Kan, He, Shan & Chen (2017).

3.3.1.2 Architectures

• **Facial expression recognition:** We evaluate our method with an inflated VGG-16 model (Carreira & Zisserman, 2017; Simonyan & Zisserman, 2015), as presented in Section 2.1.3. Prior to inflation, the CNN was pretrained with VGG-Face (Parkhi, Vedaldi & Zisserman, 2015) and FER2013 image datasets[1]. In order to facilitate comparison with the literature on AFEW, we

[1] https://github.com/XiaoYee/emotion_classification

also experiment with a C3D model (Tran *et al.*, 2015), pretrained with Sports-1M (Karpathy *et al.*, 2014).

• **Action recognition:** Because of the relatively important size of training datasets, 3D models define the state of the art on action recognition. Pretraining remains an important factor. It can be leveraged from 2d or 3d datasets. We experiment with two state-of-the-art architectures, namely Inception-3D and ResNext-101. We use the original Inception-v1 I3D (RGB only) model[2] pretrained with ImageNet and Kinetics as provided by Carreira & Zisserman (2017)[3]. The second model is ResNeXt-101 (Hara *et al.*, 2018), as presented in Section 2.1.1. Both models are very heavy to train. So we benefit from pre-training, but even the simple task of fine-tuning requires significant computational resource. From Carreira & Zisserman (2017), the I3D leveraged "synchronised distributed training with batch size 96 across 16 GPUs (each processing 6 clips)". This allows them to use clips of 64 frames (still only 2.5s of video). The ResNeXt-101 of Hara *et al.* (2018) was trained with more limited resources, with 8 GPUs, and use clips of 16 frames.

3.3.1.3 Training details

Experiments are implemented with PyTorch (Paszke, Gross, Massa, Lerer, Bradbury, Chanan, Killeen, Lin, Gimelshein, Antiga, Desmaison, Köpf, Yang, DeVito, Raison, Tejani, Chilamkurthy, Steiner, Fang, Bai & Chintala, 2019) on Compute Canada clusters (with Tesla V100 and P100 GPUs). FER models are trained on a single GPU, while action recognition models are trained with two GPUs (with batch size 16). We use the cross-entropy loss. Optimization is performed with standard SGD with momentum 0.9. We used early stopping on the validation loss. Training clips of 16 frames are selected uniformly (baseline) or with softmax sampling. As we use fairly short training clips, we remove temporal stride in convolution and pooling throughout the networks. Temporal pooling is performed at the end of the network in two stages: within clips and across clips. The fully-connected layers produce temporal score vectors of size $L \times C$.

[2] https://github.com/hassony2/kinetics_i3d_pytorch

[3] https://github.com/deepmind/kinetics-i3d

These are first max-pooled with kernel size F and stride 1 (during training $L = F$, the model produces a single vector of C class scores for each sample). For inference, we then apply softmax weighted pooling with a fixed temperature to aggregate clip-level scores.

Gradients are accumulated across batches and we perform back-propagation every 4 steps. We stop training when validation accuracy hasn't improved for 10 epochs. Experiments are reproduced several times with different random seeds. We report video-level top-1 accuracy. We use data augmentation consistently across frames[4] for a given video, with horizontal flip, random rotation, crop and color jitter.

3.3.2 Ablation Study

3.3.2.1 Illustrative examples

Figure 3.2 displays the sampling map and distribution of temporal stochastic softmax sampling, with two training videos from the AFEW dataset. The proposed method is shown to quickly identify the emotion intensity distribution in the video, thanks to the exploration steps at the beginning of training, and focuses on training clips with the best scores. As observed on the sample images, the video-level label "Happy" does not correspond to the beginning and end of the videos, because of occlusion and neutral state respectively. In both cases, the model avoids these uninformative frames.

Figures 3.3 to 3.8 provide illustrative examples, for the AFEW dataset, of the temporal stochastic softmax training process. They report sampling distributions and the temporal positions of the sampled clips at each epoch of training for a specific training video, and compare the uniform and softmax sampling strategies. The model is able to estimate meaningful distributions that correspond to the observable emotion or pain level. These distributions are built from evaluating short training clips online, iteratively throughout the training process.

[4] https://github.com/hassony2/torch_videovision

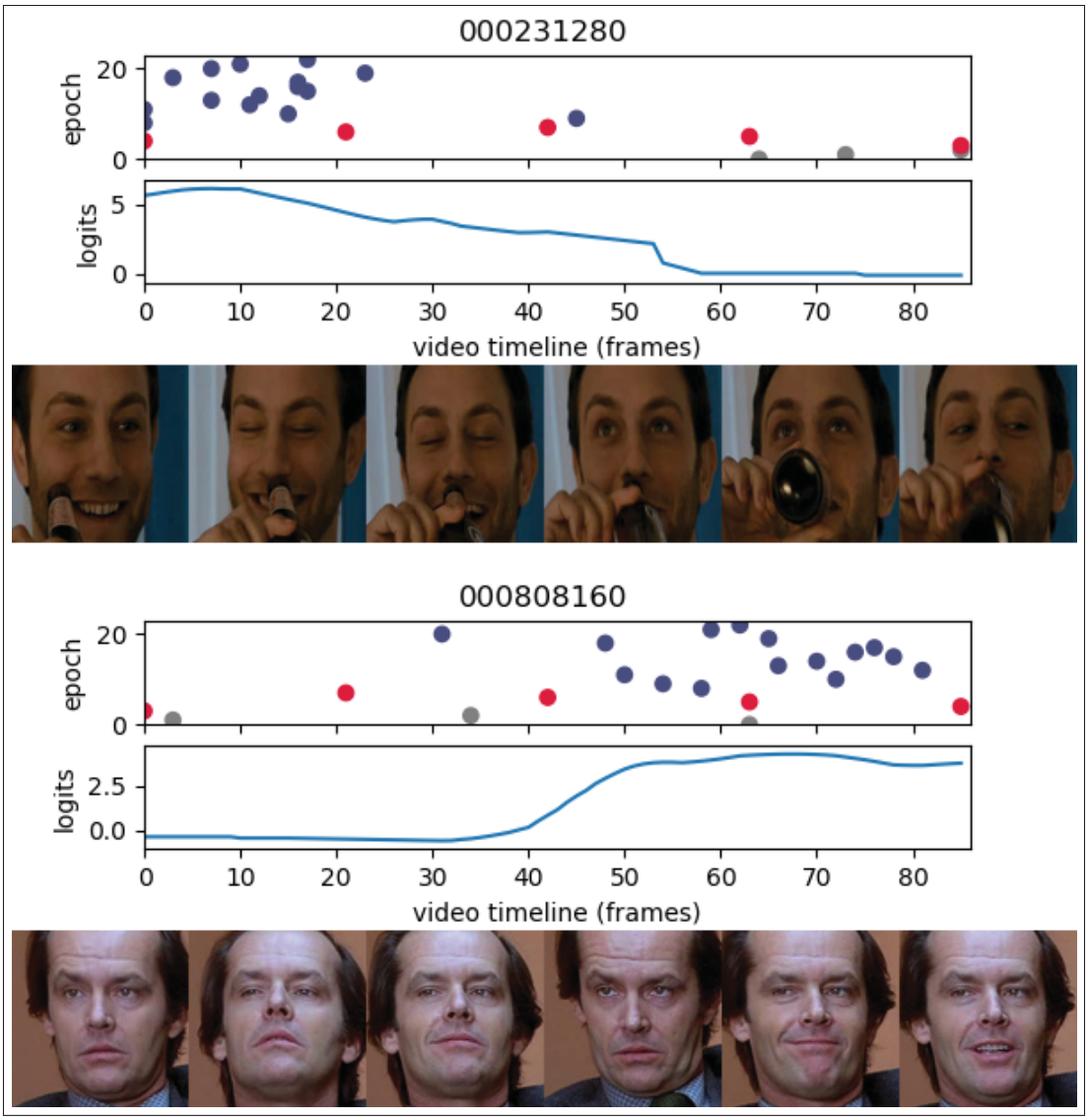

Figure 3.2 Visualization of sampling distributions (logits) and resulting clip selection during softmax training with temperature $\gamma = 1$, for AFEW videos of the *Happy* category. Sampling maps indicate the temporal position of the selected clip at each epoch for a given video example. Colors indicate the training phase: uniform warm-up, deterministic exploration, and weighted sampling, © 2021 IEEE

On BioVid, Figure 3.9 shows that the model is able to learn different expressions of pain. From the visualization of distributions obtained for *No Pain* samples of BioVid (*BL1*), it is clear that the logits are too small to provide any real advantage over uniform sampling and average pooling for these neutral states.

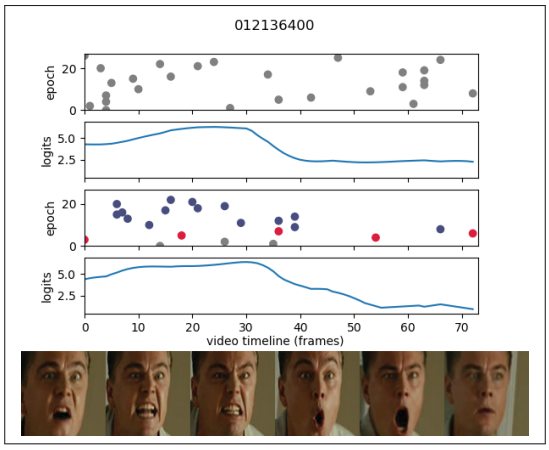

Figure 3.3 Visualization of sampling distributions for uniform training (above) and softmax temperature 1 (bellow), for *Angry* sample *012136400*

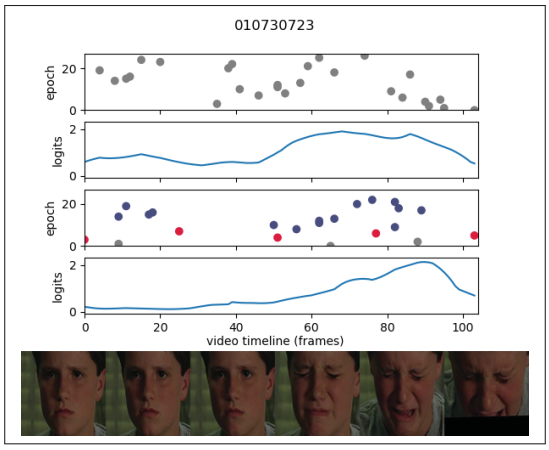

Figure 3.4 Visualization of training for sample *010730723* of the *Sad* category, with emotional progression from neutral to sad, and occlusion in the final frames

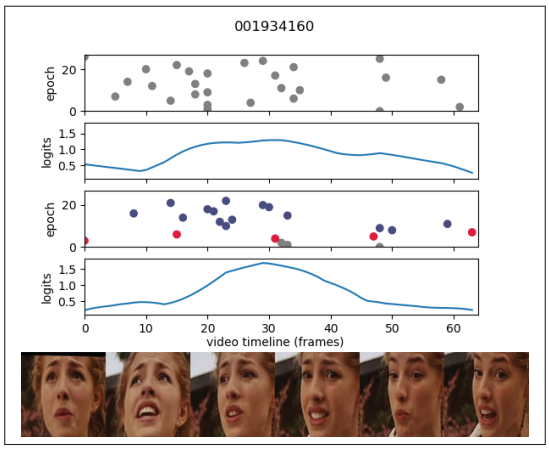

Figure 3.5 Visualization of training for sample *001934160* of the *Disgust* category, focusing on the least ambiguous expressions in the center

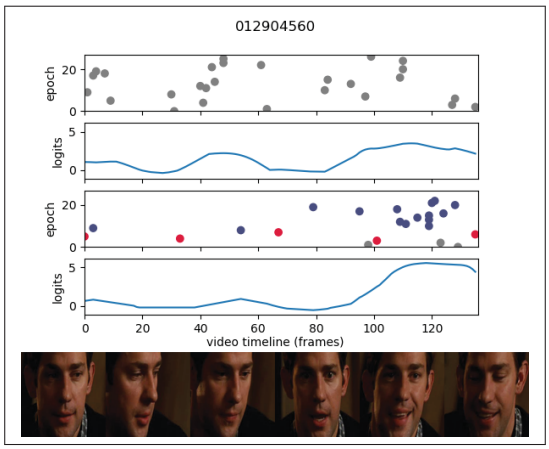

Figure 3.6 Visualization of training for sample *012904560* of the *Happy* category, avoiding neutral and surprise expressions to focus on the *happy* frames

We can note the general difference in the prediction score (logit) intensities between the datasets. This is probably due to the weight initialization of the model, which involves pretraining on 2D emotion recognition. The temperature parameter can be adapted to task-specific distributions of logits to obtain the desired sampling strategy.

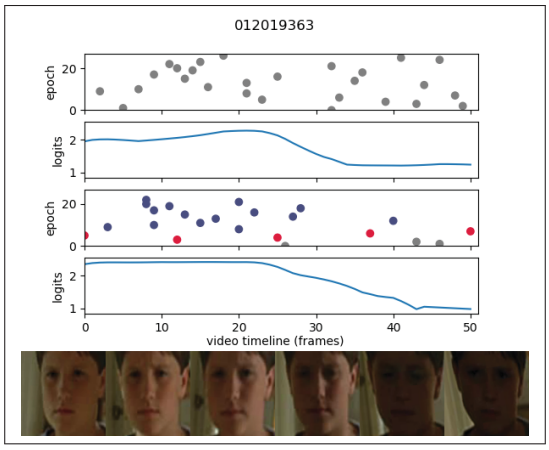

Figure 3.7 Visualization of training for sample *012019363* of the *Neutral* category, with poor lighting conditions at the end

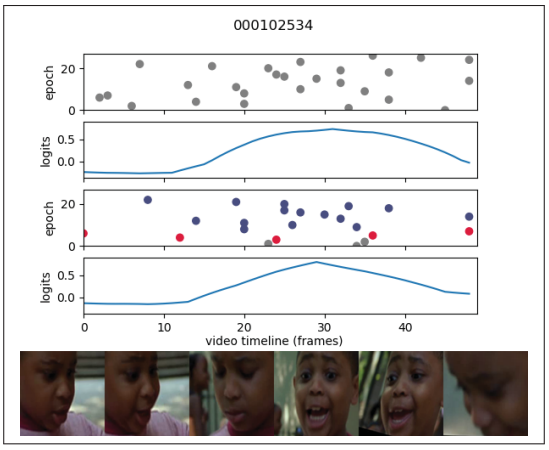

Figure 3.8 Visualization of training for sample *000102534* of the *Surprise* category, with clear apex in the middle, and head-pose variations

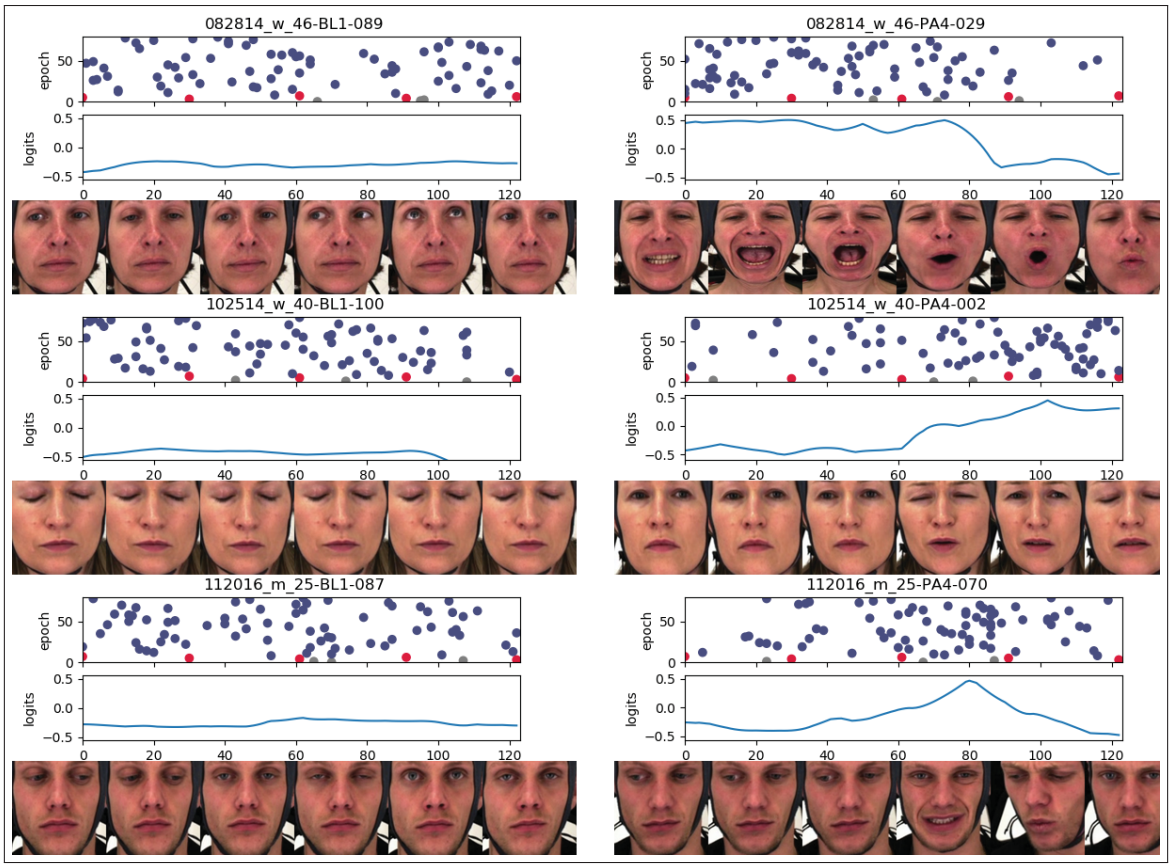

Figure 3.9 Visualizations of stochastic softmax training for three subjects of the BioVid dataset for *No Pain* (*BL1* on the left) and *Pain* samples (*PA4* on the right). The inverse temperature parameter is set to $\gamma = 2$. For neutral examples, the logits are small and negative, resulting in almost uniform sampling even with high values of γ

Table 3.1 Average performance and training duration of REINFORCE and stochastic softmax on AFEW. Both methods correspond to uniform sampling (baseline) when $\gamma = 0$, © 2021 IEEE

Inverse Temp.	**REINFORCE**		**Ours Softmax**	
	Acc. (%)	**Epochs**	**Acc. (%)**	**Epochs**
$\gamma = 0$	45.66 ±0.21	24.6	45.66 ±0.21	24.6
$\gamma = 0.5$	46.09 ±0.41	23.8	46.07 ±0.27	23.6
$\gamma = 1$	46.80 ±0.63	22.5	**47.35** ±0.27	20.3
$\gamma = 10$	44.52 ±0.18	17.5	46.65 ±0.40	17.2

3.3.2.2 Stochastic softmax sampling strategies

Table 3.1 compares results for stochastic softmax training and REINFORCE sampling. Rank-1 accuracy and the number of epochs needed to converge are presented when varying the softmax temperature. Inverse temperature $\gamma = 0$ corresponds to uniform sampling, as generally used in previous approaches. Higher inverse temperatures tend to approximate max pooling. As expected the best results are found in between average and max pooling. This shows that using a softmax pooling is important to achieve optimal performance on this task. A value of $\gamma = 1$ provides the best performance for REINFORCE as well as for our proposed method. However, the proposed softmax training manages to obtain better performance than REINFORCE because it has a lower variance in the estimation of p. The noise issue of the REINFORCE strategy is illustrated in Figure 3.10, with slow and localized updates. With softmax training, the number of epochs needed to converge for the best temperature is reduced by around 20% compared to uniform sampling. Focusing on the most important parts of the video not only improves the accuracy, it also allows the model to directly focus on important clips, thereby saving training time.

3.3.2.3 Sampling with frame-level labels

As the UNBC-McMaster dataset provides expert-annotated PSPI scores, measuring pain intensity at each frame, it can constitute an alternative to our estimated sampling distributions. Table 3.2

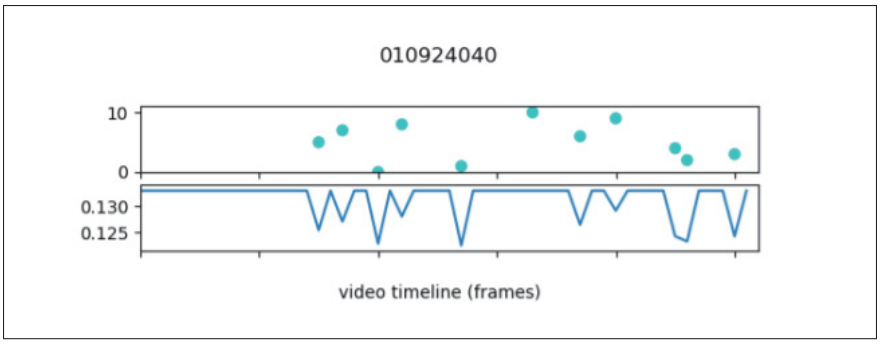

Figure 3.10 Distribution updates obtained with REINFORCE are small and localized. The distributions take too much time to fit for the needs of training

Table 3.2 Additional results of a 3D CNN on UNBC-McMaster, we compare, from top to bottom, the baseline (uniform training and average pooling), temporal stochastic softmax ($\gamma_s = \gamma_p = 2$), a decoupled version of temporal stochastic softmax ($\gamma_s = 2$ and $\gamma_p = 0$), and an experiment involving expert frame-level labels to guide sampling

Method	**EER Acc. (%)**	**Epochs**
Our baseline 3D VGG (unif.)	86.58	43.0
Stochastic Softmax ($\gamma = 2$)	87.21	37.4
Sampling only ($\gamma_s = 2$)	87.63	35.2
PSPI sampling ($\gamma_s = 0.8$)	87.84	25.8

reports the performance of a model trained with short-clips sampled with the PSPI distributions. As the PSPI scores range from 0 to 16, much higher than the classification scores produced by the model, we use a lower temperature $\gamma_s = 0.8$. With an accuracy of 87.84%, this model performs better than the weakly-supervised model. The improvement is quite limited, confirming that stochastic softmax is able to estimate meaningful distributions from sequence-level labels only. Figure 3.11 provides more details to compare PSPI and weakly-supervised sampling. The distributions estimated from sequence-level classification scores have clear similarities with the PSPI curves. We see that a data-oriented sampling strategy can replace the need for more labels. Theoretically the proposed method could also learn distributions for *No Pain*, while PSPI scores are generally zero for this class, but this doesn't seem to be relevant in our experiments.

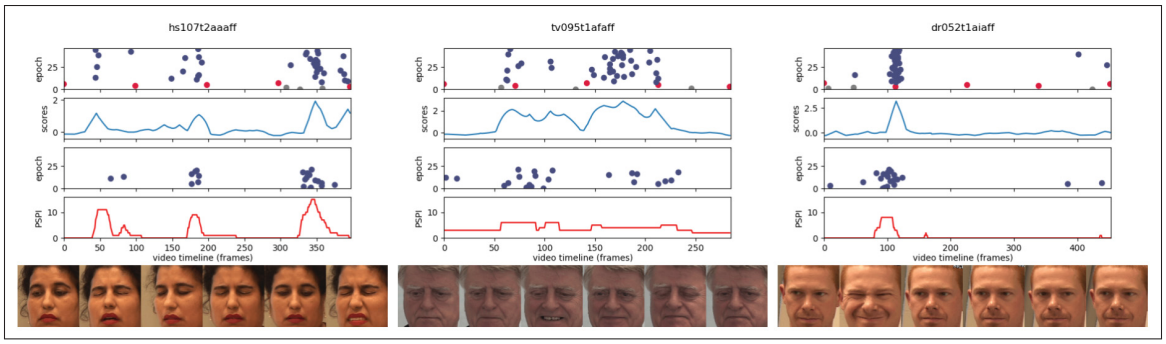

Figure 3.11 Visualizations of stochastic softmax training for three *Pain* samples of the UNBC-McMaster dataset. For each sample, the figures describe, from top to bottom, the sampling maps (temporal location for each epoch) and corresponding temporal sampling distributions, for softmax training from sequence-level labels (OPI) versus frame-level annotations (PSPI). The sequence-level label of each video is OPI 3 for *hs107t2aaaff* and *tv095t1afaff*, OPI 5 for *dr052t1aiaff*

Also, Werner, Al-Hamadi, Limbrecht-Ecklundt, Walter, Gruss & Traue (2017a) discussed limitations of the Prkaching and Solomon Pain Intensity (PSPI) scores, how they do not always correspond to pain expressions, and how their high temporal resolution might be misleading. This suggests that using expert annotations are not necessarily the best approach, even when they are available. A data-driven approach could be more efficient. However, a clear advantage of the PSPI-based sampling is the possibility to train with high intensity sampling directly, without exploration. In our experiments, this translates into a reduction of training time from 35.2 to 25.8 epochs on average.

3.3.2.4 Discussion on sampling temperatures

To evaluate the benefits of softmax weighting, we study the impact of the joint sampling and pooling temperature parameter on the classification performance and training time. Results on AFEW are reported in Table 3.3. The baseline is the uniform training, with average pooling during inference. This is equivalent to the temperature $\gamma = 0$ in the softmax framework. However, our method comprises warm-up and exploration as described in the implementation discussion, so we report both the results with uniform sampling and with $\gamma = 0$. We can see that deterministic exploration seems to be beneficial as it provides more variations in training data,

Table 3.3 Results obtained by decoupling training (sampling γ_s) and testing (pooling γ_p) softmax temperatures on the AFEW dataset. Models are trained with the clip-sampling strategy indicated in the left column, and results are provided for tests with average ($\gamma_p = 0$) and max ($\gamma_p = 10^6$) video-level temporal pooling. The *uniform* entry and $\gamma_s = 0$ differ because of the deterministic exploration at the beginning of training

Training γ_s	**Accuracy (%)**	**Epochs**	**Test $\gamma_p = 0$**	**Test $\gamma_p = 10^6$**
Uniform	45.66 ±0.21	24.6 ±2.8	45.66	46.91
$\gamma_s = 0$	46.07 ±0.20	25.7 ±3.1	46.07	46.86
$\gamma_s = 0.5$	46.07 ±0.27	23.6 ±3.1	45.61	47.00
$\gamma_s = 1$	47.35 ±0.27	20.3 ±1.7	46.59	**47.55**
$\gamma_s = 10$	46.65 ±0.40	17.2 ±2.2	45.84	46.76

but consequently delays convergence by acting as a regularizer. Weighted sampling is effective after the 8^{th} epoch (3 epochs of warm-up and 5 epochs of exploration). With large γ temperature parameters, the best clips are exploited more often, leading to lower training data variation and more informative clips. We observe a shortening in the duration (epochs) of the training process. The best classification performance (47.35%) is obtained with softmax temperature $\gamma = 1$, which provides a compromise between uniform and maximum temperatures, effectively focusing on relevant clips while maintaining diversity in selection. Results with softmax temperatures during training and average pooling during testing (Table 3.3, Test $\gamma_p = 0$) demonstrate the effect of sampling temperature on the learning phase. Interestingly, we observe that having different temperatures during training (clip sampling γ_s) and during testing (temporal pooling γ_p) can lead to even better performance. When considering this possibility, the best accuracy (47.55%) is obtained with $\gamma_s = 1$ for softmax training-clip sampling, and $\gamma_p = 10^6$ for max pooling. The *Sampling only* experiment on UNBC-McMaster reported in Table 3.10 also supports this hypothesis. It shows that weighted sampling improve training quality on its own. Although here, the average pooling at test time seems more efficient than max pooling, probably due to the differences between the categorical emotion recognition task and the binary pain detection setup (as discussed in the Results section of the paper). As we designed this method as a unifying framework, we do not extensively study the effect of decoupling the two softmax temperatures.

Table 3.4 Accuracy on HMDB-51 of the ResNext-101 model with stochastic softmax sampling and softmax pooling

Inverse Temperature	**Accuracy (%)**	**Epochs**
Uniform	58.13 ±1.00	40.0 ± 2.1
$\gamma = 0$	58.63 ±0.89	36.0 ± 2.9
$\gamma = 0.25$	59.48 ±0.71	44.3 ± 4.9
$\gamma = 0.5$	59.74 ±1.05	52.0 ±12.2
$\gamma = 1$	57.95 ±1.17	44.7 ±16.2
$\gamma = 10$	56.98 ±1.38	30.7 ± 5.8

Table 3.5 Accuracy on HMDB-51 of the Inception-3D model for stochastic softmax sampling with softmax or average pooling

	Softmax sampling and pooling $\gamma_p = \gamma_s$		**Softmax sampling only** $\gamma_p = 0$	
Inv. Temp.	**Accuracy (%)**	**Epochs**	**Accuracy (%)**	**Epochs**
Uniform	66.79 ±0.29	86.7 ± 4.1	66.79 ±0.29	86.7 ± 4.1
$\gamma_s = 0.25$	**67.37** ±0.31	87.0 ± 9.2	66.93 ±0.17	87.5 ± 4.4
$\gamma_s = 0.5$	66.75 ±0.27	85.3 ± 4.6	67.51 ±0.23	87.8 ± 2.5
$\gamma_s = 1$	65.50 ±0.34	102.4 ± 6.5	67.50 ±0.11	103.6 ± 9.1
$\gamma_s = 5$	64.38 ±0.28	85.9 ± 5.0	**67.75** ±0.19	98.3 ± 5.5
$\gamma_s = 10$	64.35 ±0.33	80.5 ± 4.2	67.23 ±0.18	88.9 ± 3.5

As seen in Tables 3.4 and 3.5, the training temperature also shows beneficial on the action recognition task. However, the improvement is less pronounced. The models are pretrained on a very similar task and using softmax sampling when training from scratch could probably reveal more efficient than with fine-tuning only. The optimal temperature seems to be lower than with AFEW. The reason might be inherent to the dataset, with videos having different length and information distribution. For temporal pooling, average is the best strategy here, and softmax harms the recognition performance (Table 3.5). The model only benefits from the training strategy. Sampling behaviors are illustrated in Figures 3.12, 3.13, and 3.14.

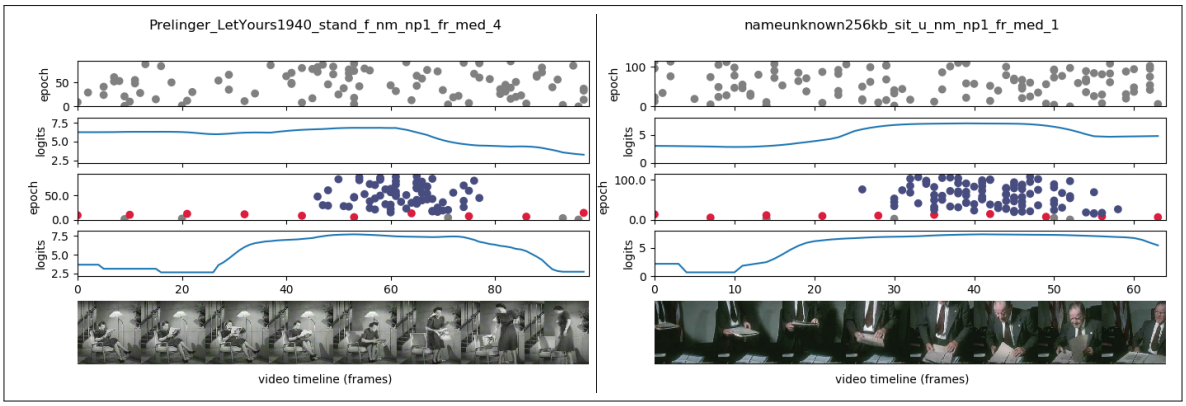

Figure 3.12 Visualizations of sampling distributions for uniform training (above) and softmax temperature 10 (bellow), for videos of action categories *stand* (left) and *sit* (right), with clip sampling focused on the actions

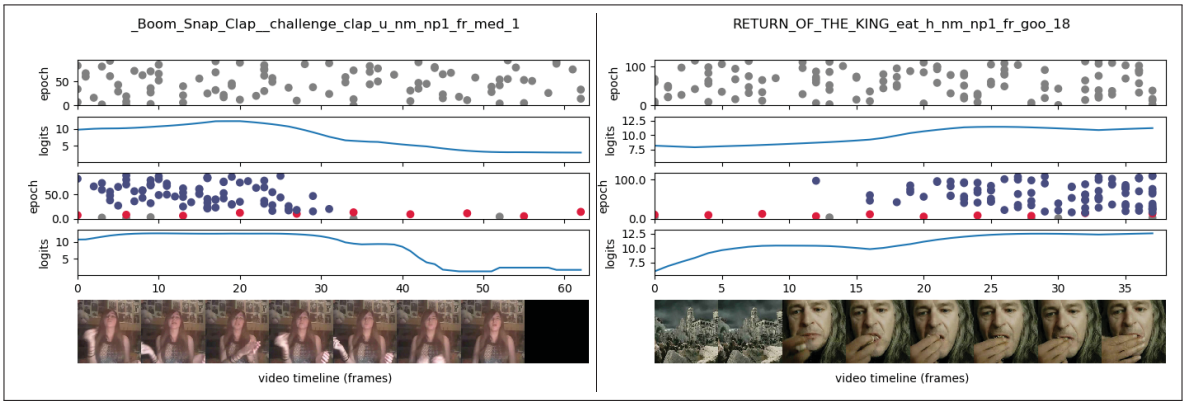

Figure 3.13 Visualizations of sampling distributions for uniform training (above) and softmax temperature 10 (bellow), for examples of categories *clap* (left) and *eat* (right), with occluded and irrelevant frames identified at the end and beginning of the videos respectively

3.3.2.5 Clip duration

We study the effect of training-clip duration on classification accuracy, for different temperatures of sampling. We perform very limited hyper-parameter search for this study, so performances could probably be improved for large values of clip duration. Results presented in Table 3.6 show that accuracy improves with clip size, but the impact of stochastic softmax is greater for smaller clips. This confirms our initial hypothesis. Uniform sampling ($\gamma = 0$) particularly

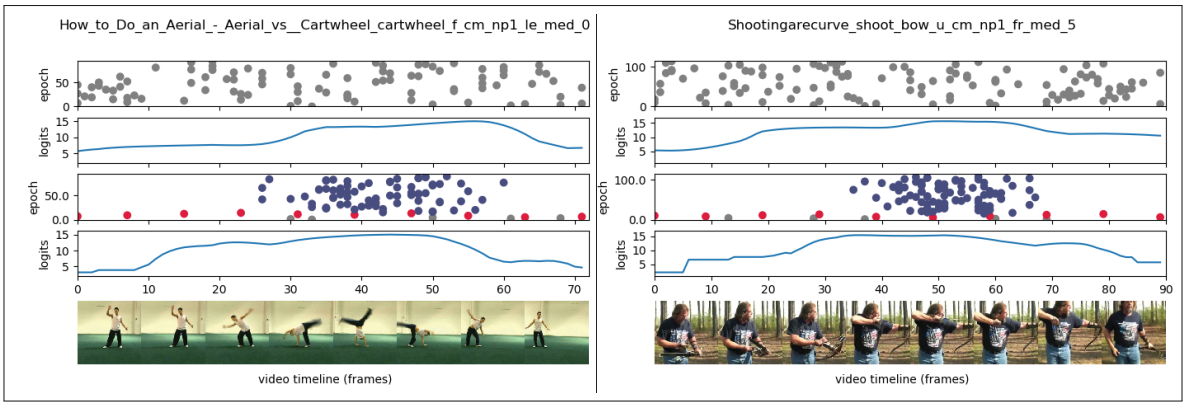

Figure 3.14 Visualizations of sampling distributions for uniform training (above) and softmax temperature 10 (bellow), for videos of the action categories *cartwheel* (left) and *shoot_bow* (right)

benefits from larger clips, as they will reduce noise in gradients and training inputs will be closer to those in inference mode (long videos). Weighted sampling on the contrary has more impact with small clips, as they allow for more precise focus and avoiding of irrelevant clips. Indeed, max and average sampling are equivalent when the clip contains the entire video. Note that all clips become similar when their size is large, with more overlapping frames.

For HMDB-51, we can see a sweet spot emerging from these two trends. As show in Table 3.7, the best configuration is clip size 16 with $\gamma = 0.25$. Again, increasing clip size requires to reduce batch size, so the evaluation is biased, but this describes the practical phenomenon related to computational constraints that is part of our study.

Table 3.6 Influence of training-clip duration for classification accuracy (%) with 3D-CNN stochastic softmax on AFEW

Inverse Temperature	**Clip duration (frames)**		
	8	**16**	**32**
$\gamma = 0$	45.17	45.78	47.09
$\gamma = 1$	46.39	47.00	**47.43**
$\gamma = 10$	46.04	45.85	47.17

Table 3.7 Influence of clip size with ResNeXt-101 model, for split 1 of HMDB-51

Inv. Temp.	Clip duration (frames) 8	16	32	64
$\gamma = 0$	54.71	59.15	**60.72**	59.08
$\gamma = 0.25$	54.84	**61.70**	60.00	58.37
$\gamma = 0.5$	**56.73**	60.59	59.48	**59.74**
$\gamma = 1$	52.09	56.54	59.74	59.15
$\gamma = 10$	50.52	57.39	59.61	58.69

3.3.2.6 Clean sample scoring

In order to update the sampling distributions, it is straightforward to use the scores obtained during training. However, these scores are subject to data augmentation and dropout. This introduces noise in the estimation of temporal distributions. Table 3.8 shows this phenomenon. Using "clean" copies of the clips to evaluate their score makes temporal sampling more efficient. In our experiments, using softmax temperature $\gamma = 10$, accuracy was 46.65% with data-augmented samples and 46.91% with clean samples (with a uniform sampling baseline of 45.87%). The computational overhead is very limited as it consists in adding only an inference step on small clips with no back-propagation. We can note that even when taking the readily available scores to update the distributions, our sampling method performs significantly better than uniform sampling.

Table 3.8 Influence of scoring from clean samples compared to directly using the data-augmented training samples, on AFEW

Training method	Accuracy (%)
Baseline ($\gamma = 0$)	45.87
Clean scoring, $\gamma = 10$	46.91
Data-augmented scoring, $\gamma = 10$	46.65

3.3.3 Comparative results

Table 3.9 presents our results in comparison with other 3D-CNN approaches on AFEW. Our inflated VGG-16 achieves very good performance compared to other 3D CNNs. Temporal stochastic softmax is able to improve accuracy further. C3D Weighted (Vielzeuf *et al.*, 2017) performs a temporal weighting of the training-clip losses, thus a simplified approximation of our method. We also validate experimentally the effect of stochastic softmax training with a C3D, as this architecture has been extensively studied in the literature. Results show that the training method can be adapted to different architectures of 3D CNNs.

Table 3.9 Performance of 3D-CNN stochastic softmax on AFEW compared to our baseline (same architecture but uniform training-clip sampling and average pooling) and relevant literature, © 2021 IEEE

Reference	**Method**	**Acc. (%)**
Lu *et al.* (2018)	3D VGG-16	39.36
Fan *et al.* (2016)	C3D	39.69
Vielzeuf *et al.* (2017)	C3D-LSTM	43.2
	C3D Weighted	42.1
C3D baseline	C3D (uniform)	39.95
C3D with Softmax	C3D ($\gamma = 1$)	42.78
VGG baseline	3D VGG-16 (unif.)	45.66
VGG with Softmax	3D VGG-16 ($\gamma = 1$)	47.35

Table 3.10 EER-Accuracy of a 3D CNN on UNBC-McMaster, with and without stochastic softmax, compared to related literature, © 2021 IEEE

Reference	**Method**	**EER Acc. (%)**
Wu *et al.* (2015)	MIL-HMM	85.2
Werner *et al.* (2017a)	SPTS+CAPP	91.7*
Sikka & Sharma (2018)	LOMo with SIFT, LBP	87.0
Kumawat, Verma & Raman (2019)	LBVCNN	86.55
Our baseline	3D VGG (unif.)	86.58
Stochastic Softmax	3D VGG ($\gamma = 2$)	87.21

* Accuracy is not computed at ROC-EER.

Table 3.11 Binary classification accuracy of a 3D CNN on BioVid (*BL1* vs. *PA4*), with and without stochastic softmax, compared to related SOTA methods, © 2021 IEEE

Reference	**Method**	**Acc. (%)**
Othman, Werner, Saxen, Al-Hamadi & Walter (2019)	RFc with FADs	65.8
	Reduced MbNetV2	65.5
Thiam, Kestler & Schwenker (2020)	Two-stream VGG	69.25
Werner *et al.* (2017a)	Standardized FADs	72.4*
Our baseline	3D VGG (uniform)	68.12
Stochastic Softmax	3D VGG ($\gamma = 2$)	69.60

* Using additional information with depth sensor 3D maps.

Table 3.12 ROC-AUC results on BioVid (*BL1* vs. *PA3-4*), with and without stochastic softmax, compared to related literature, © 2021 IEEE

Reference	**Method**	**AUC (%)**
Tavakolian & Hadid (2019)	3D ResNet	82.54
	S3D-G	83.26
	SCN	86.02
Our baseline	3D VGG (uniform)	82.67
Stochastic Softmax	3D VGG ($\gamma = 2$)	84.39

Tables 3.10, 3.11 and 3.12 report the performance of our sampling method on three pain video classification tasks. An aggressive sampling temperature could be expected to generate overfitting on UNBC-McMaster, by reducing the variation in training data which is already very limited. Actually, we found that better results were obtained with a higher temperature, $\gamma = 2$ (Table 3.10). The classification scores of the model on UNBC-McMaster are generally lower than for AFEW. The temperature parameter allows us to adapt the weighting strategy. Note that with short-clip training, the model cannot adapt its scores to learn a video-level temperature implicitly. Table 3.11 provides results on the larger BioVid dataset. Temporal max-pooling during inference might not be beneficial, because it is necessary to consider the entirety of the video to classify it. For example, the model could recognize a neutral expression at the beginning of a Pain video, with very high confidence. We considered decoupling the temperatures of the

sampling and pooling but do not extensively study this possibility here. The good performance improvement obtained with our softmax weighting on BioVid (Table 3.11) suggests that the method is also relevant to very controlled recordings, with no occlusion and very limited head movement. The benefits of clip sampling does not only consist of limiting the impact of noisy labelling or capture conditions. Temporal weighting addresses a challenge inherent to the task, as facial expressions are events localized in time, and typically preceded and followed by neutral states. This idea is confirmed as the best performance gain of softmax weighting is observed on the second task of BioVid (Table 3.12). In this scenario, additional samples are gathered in the *Pain* class (*PA3* + *PA4*). These are videos where temporal weighting is really efficient, as opposed to *No Pain* videos which do not contain apex or particularly relevant segments. This calls for future work studying the relevance of using different softmax temperatures adapted to each class, as was developed by McFee *et al.* (2018) with temporal pooling for audio data.

3.3.4 Discussion

We presented a softmax-based training and inference method for 3D CNNs adaptable to the task at hand in terms of computation, regularization and feature aggregation strategy, with no additional trained layer. Our method is designed to enable the use of softmax temporal pooling within the framework of short-clip training, which is the standard way of training 3D models because of computational and GPU-memory limitations. We demonstrated the benefits of softmax temperatures for video classification by considering videos as bags of unequally relevant clips. At test time, a temporal softmax pooling mechanism is able to weight and aggregate information from different clips, with a strategy adapted to the input distribution. Softmax-weighted stochastic sampling improves learning by balancing informative and difficult training clips, allowing for faster convergence and limiting the impact of irrelevant clips in the context of weak, sequence-level annotations. These mechanisms provide an improvement in accuracy on all evaluated datasets. Experiments suggested several directions for future work.

CHAPTER 4

SPATIO-TEMPORAL ATTENTION MECHANISMS

This chapter studies spatiotemporal visual attention, and how to integrate it into 3D CNNs, specifically for the task of emotion recognition from facial expressions, working with aligned face image sequences. Attention has been a hot topic in deep-learning research for the past few years. There have been examples of very good performances. Attentional behavior, focusing on specific regions of data, can have several advantages. Performance can be improved by reducing the noise introduced by irrelevant information. Also, by only analysing informative regions, the computational cost of classification can be reduced.

Attention has been successfully leveraged across many applications, as neural machine translation (Bahdanau *et al.*, 2015; Vaswani *et al.*, 2017), localization and understanding in images (Jaderberg *et al.*, 2015; Ba *et al.*, 2015), image captioning (Xu *et al.*, 2015), image generation (Gregor *et al.*, 2015; Zhang, Goodfellow, Metaxas & Odena, 2019), or video understanding (Girdhar *et al.*, 2019; Aminbeidokhti *et al.*, 2019; Li *et al.*, 2018a).

The notion of attention regroups a lot of different ideas, mechanisms and behaviors. In Section 3.1, we observed that our stochastic softmax sampling method shares similarities with attention mechanisms. We now study attention in detail. The next section provides key concepts to understand attentional behaviors. Popular archetypes of attention mechanisms are then presented. They are mask attention, transformation networks and self-attention. Theoretical capabilities and benefits of each type of attention is discussed, as well as the modifications that have been proposed in literature. The focus of this study is the adaptation or improvement of these modules to spatiotemporal dimensions, for integration in 3D CNNs for video FER. Problems raised by spatiotemporal attention, and specifically with 3D models are highlighted. This encompasses issues inherent to spatiotemporal data (e.g. high number of positions in three dimensional space), and issues related to the task's implementation (e.g., images are already aligned to faces spatially but not temporally). Experiments are then reported for these types of

attention mechanisms, with ablation studies notably for different attention dimensions, temporal, spatial and spatiotemporal.

4.1 Definition of attention

In Section 1.2.6, we gave an overview of what attention is and can achieve. The next sections focus on typical attention modules, discussing their benefits and relevance to spatiotemporal models and facial expression recognition. Attention mechanisms for deep neural networks are inspired from human visual attention (Posner & Dehaene, 1994; Corbetta & Shulman, 2002; Petersen & Posner, 2012; Itti, Koch & Niebur, 1998; Larochelle & Hinton, 2010). Several types of attention mechanisms can be developed, producing a wide range of behaviors.

• **Focus:** The principle of attention is to focus on salient regions of the input, as human decide where to look, with a limited field of view around a fixation point. In computer vision, we can distinguish two ways for attention mechanisms to implement this concept: hard attention and soft attention. With hard attention, the field of view is extracted from the input with an actual cropping operation (Mnih *et al.*, 2014; Korbar *et al.*, 2019; Ba *et al.*, 2015; Sermanet *et al.*, 2015; Gregor *et al.*, 2015; Jaderberg *et al.*, 2015). The attended region will be processed with high resolution, while other regions will be treated less precisely or skipped entirely. This idea was particularly studied in the work of Larochelle & Hinton (2010), with a foveal representation of the attended regions. Sampling a region of the input with a spatial crop reduces the size of the processed data, and thus the computational cost of feature extraction. However, focal attention usually requires an iterative process, aggregating information across several glimpses. Also, by focusing on relevant regions, the model avoids processing background information, and improves the signal-to-noise ratio on the processed data. The authors of Hu, Shen & Sun (2018) summarize this idea: “Attention can be interpreted as a means of biasing the allocation of available computational resources towards the most informative components of a signal”.

Soft-attention mechanisms remove the localization constraint of focal attention, enabling real-values (as opposed to binary) of attention intensities spread across space. This allows for much

more precise and adapted attention profiles (Bahdanau *et al.*, 2015; Xu *et al.*, 2015; Vaswani *et al.*, 2017; Li *et al.*, 2018a; Woo *et al.*, 2018). Instead of cropping a spatially continuous region of the input, regions are inhibited or enhanced with an attention mask. Contrary to hard attention, this does not provide any reduction in computational requirements.

• **Orthogonal computation:** Within deep learning neural networks, this behavior is achieved with attention mechanisms that do not follow the feed-forward flow of the main feature extraction network, similarly to how the attention system of human vision is anatomically different than those who process the attended information (Posner & Dehaene, 1994). Schematically, the attention mechanism is orthogonal to the main branch (which produces classification scores) and controls the information flow. This is obvious with sampling-based hard-attention mechanisms, searching for features that will guide the position of the sensor in the attended scene, but will not directly be used to build high-level representations for the classifier. But for soft attention mechanisms, the frontier between attention and feature extraction is not well defined.

This type of operation is also found in gating mechanisms for adaptive inference graphs of neural networks (Bengio, Bacon, Pineau & Precup, 2015; Veit & Belongie, 2020), or controlling information flow in memory cells of RNN (Hochreiter & Schmidhuber, 1997; Cho, van Merrienboer, Gülçehre, Bahdanau, Bougares, Schwenk & Bengio, 2014), or with gating mechanisms implementing attention over channels or features dimension (Miech, Laptev & Sivic, 2017; Hu, Shen, Albanie, Sun & Wu, 2020; Bhuiyan, Liu, Siva, Javan, Ayed & Granger, 2020).

• **Contextual attention:** Studies on human vision have shown that attention is task dependent (Hayhoe & Ballard, 2005; Mathe & Sminchisescu, 2013). Objectives and goals influence the positioning of eyes and fixations. The study of Ba (2020) gives a functional definition of visual attention: “attention allows for salient features to dynamically come to the forefront as needed”. The idea of dynamic process is related to the use of a context or query that conditions the feature selection. The hard attention mechanism selects a location where to look at, based on information accumulated from previous glimpses, and contextual information representing the

entire image with low resolution. The model can use its current hidden state to know what to look for, and the context information to know where to find it.

In this sense, the attention mechanism implements an objective in the feature extraction process, with a way for the model to specify what it is looking for in the input (at a given step or for a specific task), and to decide where to find it. This behavior has been implemented with different approaches and for different dimensions (spatial, temporal and more). Torralba, Oliva, Castelhano & Henderson (2006) discuss how global scene context guides eye movements and attention. For caption generation (or machine translation on text) attention provides a way of conditionning the ouput on some previous state in a sequential generation process (Bahdanau *et al.*, 2015; Xu *et al.*, 2015; Vaswani *et al.*, 2017). For each step, an attention map is produced to weight the input positions (pixels in images or tokens in a text sequence). This attention network is parameterized with the hidden state of the generator (formally this context vector is an input to the attention function), to provide a different attention profile for each step of the generation process.

Methodology: Attention mechanisms are compared based on the performance gain they provide (compared to the baseline model without attention). We evaluate the cost of the additional layers and computations required by the attention module. We describe the expected behavior, and the theoretical abilities of the mechanisms. As the modules are trained end-to-end and learn from data, we visualize the attention vectors to validate and describe the observed behavior.

4.1.1 Links with memory addressing

Another related research domain concerns external memories for neural networks, usually on the task of question answering, with the design of differentiable memory-addressing mechanisms (Graves, Wayne & Danihelka, 2014; Sukhbaatar, Szlam, Weston & Fergus, 2015; Cheng, Dong & Lapata, 2016). The operations of reading from and writing to each element of the memory are modeled by a neural network, through a soft attention vector. These mechanisms are similar to intra-attention and compute compatibility or similarity between entities. In the

Neural Turing Machines of Graves *et al.* (2014), the read operation is a weighted average of the $M \times N$ memory (N locations, M feature channels). A weight vector of N values is produced and normalized, to weight the impact of each location on the read operation. This content-based addressing (as opposed to position based) focuses attention on locations based on the similarity between their current content and a vector emitted by the controller (that can be interpreted as a query). This is similar to a mechanism of visual attention, in which an object is focused on because its visual features correspond to what is looked for, and not because of its position. The difference between location-based and content-based attention for neural machine translation is discussed in Luong, Pham & Manning (2015).

A hard-attention version of Neural Turing Machines is even proposed by Zaremba & Sutskever (2015), working with non-differentiable memory addressing for efficient and scalable query engines.

4.1.2 Sequential mechanisms of visual attention

The Recurrent Attention Model (RAM) proposed in Mnih *et al.* (2014) and further studied by Ba *et al.* (2015) and Sermanet *et al.* (2015) is based on the human visual attention, gathering information from different locations by orienting the fovea to salient objects. A single image is processed by a sequential model, accumulating information from various locations through a RNN. This idea of processing an image as a sequence of glimpses was already studied by Larochelle & Hinton (2010). This mechanism implements hard attention as it only focuses on part of the input, while other positions will not be processed. This attention behavior is learned from data, and should focus on discriminant information in space.

Developing RAM in a spatiotemporal setup would be very interesting, with 3D glimpses extracted sequentially from a video volume. However, processing glimpses requires specific, small networks, employed recurrently on the sequence of glimpses. This doesn't fit in our framework of study and cannot be compared easily with other attentions methods. It would be

closer to a part-based feature extractor (Jeni, Girard, Cohn & la Torre, 2013; Chen, Zhou, Su, Wu, She & Hirota, 2018a).

Because of the non-differentiable, stochastic nature of this sampling mechanism, RAM is trained with reinforcement learning which makes it harder to use, as stochastic gradient estimates can have high variance. Each training image is transformed into a learning environment within which the model optimizes its strategy to deploy its sensor efficiently. This explains the limited popularity RAM received in literature, other than work to improve the training method. However, we can also hypothesize advantages of learning attention with reinforcement. Reinforcement learning is by nature a form of data-augmentation, providing a large number of possible glimpse trajectories from each training image.

Alternatively, Kahou, Michalski, Memisevic, Pal & Vincent (2017) develop a recurrent attentive tracking model operating sequential glimpse sampling, but with a differentiable sampling mechanism based on a spatial grid of Gaussian filters (similarly to Gregor *et al.*, 2015). This enables end-to-end training with gradient-based learning methods.

4.2 Mask attention

"Here we consider attention as a set of spatial maps that essentially try to encode on which spatial areas of the input the network focuses most for taking its output decision (e.g., for classifying an image)". In the work of Zagoruyko & Komodakis (2017) attention is considered as simply determined by neuron activations. In this sense, all CNNs already have an attentional behaviour. Gating attention mechanisms are able to complement this behaviour with an explicit scaling of the features. This allows the decoupling of feature activations at a specific location and its impact on the final decision. The objective is to regress a valence score for each input frame or spatial position (or both).

Attention masks share similarities with the task of saliency prediction. Specialized datasets are available to train neural networks to detect regions of interest in images (Cornia, Baraldi, Serra & Cucchiara, 2016). These datasets can be used as a form of supervision for the attention

module, as in Mavani, Raman & Miyapuram (2017). Usually, the attention mechanism is trained end-to-end with the main FER network, minimizing the classification loss, without direct attention supervision.

An attention map, or attention mask, is produced by regressing a scalar attention weight α_n for each location n in the feature map f:

$$y = \alpha \cdot f. \tag{4.1}$$

Attention is applied to the input feature map with Hadamard product (element-wise multiplication), resulting in another feature map y, with $y_n = \alpha_n f_n$. This operates a gating mechanism, which inhibits or enhances feature activations independently, allowing the following layers to focus only on selected features.

Alternatively, the attention mask can be used for an attentional weighted pooling mechanism over the N locations of the feature map:

$$y = \sum_{n=1}^{N} \alpha_n f_n. \tag{4.2}$$

Softmax is often applied to the attention map to have an attention density summing to 1. This means each position competes with the others for attention. In the work of Wang *et al.* (2018b), this softmax operation is what defines an attentional behavior. However, the choice of the activation function remains an open issue, as in a lot of applications. The authors of Rahman, Rochan & Wang (2019) and (Zhao, Li, Zhuang & Wang, 2017) prefer the use of sigmoid activation for attention weights, even when the attentional weighting is followed by summation. This allows to provide an attention weight for each position independently of others.

4.2.1 Attention heads

Li *et al.* (2018a) use several attention models in parallel, using a diversity regularization to have them focus on different locations (body parts in this case). Each of these attention heads learns to identify, localise and enhance a specific type of object or feature in the image. This produces a richer attentional behavior. The gated feature vectors produced by the different attention heads are aggregated to obtain a new representation of the input.

4.2.2 Contextual attention

Instead of computing weights from each position independently (Li *et al.*, 2018a; Lin, Feng, dos Santos, Yu, Xiang, Zhou & Bengio, 2017; Woo *et al.*, 2018), contextual information can be leveraged to produce a dynamic attentional behavior. The feature representation of each position is interpreted in light of the contextual information from other positions. In the work of Rahman *et al.* (2019) and Zamprogno, Passon, Martinel, Serra, Lancioni, Micheloni, Tasso & Foresti (2019), the attention map is produced iteratively by a recurrent mechanism, which refines the attention weights, integrating video-level context into the representation of each frame. Similarly, Meng *et al.* (2019) use a two-stage temporal attention mechanism, with frame-based weighting followed by video-level context integration.

4.2.3 Spatio-temporal attention masks

Attention can be produced along any dimension, spatial, temporal or the feature channel dimension. Das, Chaudhary, Brémond & Thonnat (2019) develop a pose-guided attention mechanism over body parts for action recognition. Li *et al.* (2018a) employ a two steps attention, with spatial attention first and then temporal attention. Woo *et al.* (2018) apply attention on the channel dimension followed by the spatial dimension. Channel-wise attention was especially discussed in Hu *et al.* (2018), with Squeeze-and-Excitation Network. The objective is to model global channel inter-dependencies, which is limited in convolutional layers. Feature maps are scaled with channel-wise weights computed with a non-linear transformation of the

spatially-squeezed input. Spatial average-pooling allows to represent the image's channel-wise statistics with a single value per channel. This can then be used to compute global channel inter-dependencies.

Many works on spatiotemporal attention do not process spatial and temporal dimensions the same way. For instance Lee *et al.* (2018) use a spatial attention mask for each frame in a video. This is because using an actual attention mechanism in the temporal dimension is more difficult, because of the varying size of videos, and the clip-sampling mechanism.

Indeed, a spatiotemporal attention mask could be created with a joint softmax over both spatial and temporal dimensions. One issue consists in the number of spatiotemporal positions being very important ($L \times H \times W$ typically) and the softmax function tends toward uniform distributions then. Also, as the model is trained with short clips, attention over the entire videos cannot be learned.

4.3 Transformation networks

The Spatial Transformer module was conceptualized in Jaderberg *et al.* (2015) to make CNNs more spatially invariant to the input data. The module is explicitly designed to operate spatial transformations of images or feature maps within the network. This mechanism is differentiable and trained with no extra supervision.

The transformation applied to the feature map is conditioned on the feature map itself, and through a learned neural network. Possible transformations include scaling, cropping, rotations, and non-rigid deformations. A wide range of transformations can be applied. They only need to be differentiable with respect to the parameters. The transformation block defined for the Spatial Transformer Network (STN) does not perform feature extraction. The STN module can be considered as a differentiable hard-attention mechanism because it can operate a crop of the input, to focus following feature extraction on a specific part of the input.

In addition to the attention behavior produced by cropping the input around a selected region of interest, the authors consider spatial transformations capable of alignment of the data. Indeed, the STN module can identify pose in the input image and correct it with a transformation that produces a canonical pose, learned for the task. Such alignment facilitates pattern recognition in the following convolutional layers, and improve spatial invariance. This behavior is explicitly preferred over attention in works that exploit STN modules for motion compensation between consecutive frames of a video Caballero, Ledig, Aitken, Acosta, Totz, Wang & Shi (2017); Dai, Zhang, Wang, Lu & Wang (2019).

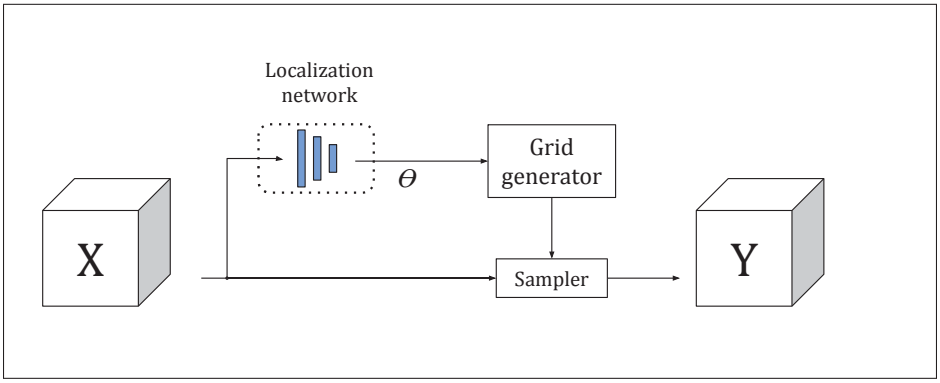

Figure 4.1 Architecture of the spatial transformer module presented in Jaderberg *et al.* (2015)

As described in Figure 4.1 (from the original work of Jaderberg *et al.*, 2015), the localization network determines the transformation parameters with a forward pass on the input feature map. These parameters are then used to generate a sampling grid which is applied to transform the feature map. Bi-linear interpolation is typically used for the sampling process. Each interpolated pixel value in the output feature map is determined by a weighted average of the closest 2×2 reference pixel neighborhood targeted by the sampling grid.

The standard type of transformations proposed is affine transformations. They can be easily be defined with 2×3 parameters for 2D data, or 3×4 parameters for 3D (e.g. spatiotemporal). They allow cropping, translation, rotation, scale, and skew to be applied to the input feature map. Also, the transformation can be constrained to even more typical behavior for attention mechanisms (cropping, translation and isotropic scaling) with only 3 parameters (4 parameters for 3D).

The authors highlight the benefits of using structured and low-dimensional parameterization of transformations to facilitate the the task of the localization network. Alternatively, the entire sampling grid can be regressed entirely from the feature map. This allows basically unconstrained transformations and increases complexity but can be useful for motion compensation between consecutive frames. In this case the the grid generator can be designed as a flow estimator, to apply a transformation that cancels motion Caballero *et al.* (2017); Dai *et al.* (2019).

With FER, as we already employ a pre-processing step of face extraction and alignment, STN might be redundant if it only rotates the image to obtain pose invariance. In this sense, we can consider FER as closer to tasks like the house number recognition in the SVHN dataset. Data is already cropped around the region of interest. There, the work of Mnih *et al.* (2014) on Recurrent Attention Models (RAM) showed the benefits of using focusing attention with cropping mechanisms if they operate in a recurrent way, producing several locations to attend to, within the already aligned picture.

Also we could use several STN attention heads looking for different ROIs in the face, e.g. based on Action Units. Then the pretrained VGG model wouldn't be working with face-aligned images, which is not a very promising approach as we already saw that pretraining is necessary because of the small size of our dataset. It would probably require more resources to train expert networks adapted to specific attention heads.

4.3.1 Spatio-temporal transformation networks

For video FER, temporal attention could be performed by a spatiotemporal transformer network. Dai *et al.* (2019) employ a STN on each frame to reduce noise and help with alignment of moving pedestrians. The STN is applied on each frame, performing a spatial transformation only. However, they use a RNN to generate the sequence of STN parameters applied to each frame. This favors continuity in the spatial transformations of consecutive frames.

Kim, Sajjadi, Hirsch & Schölkopf (2018) actually augment the STN to a STTN (Spatio-Temporal Transformer Network) working with data volumes and performing spatiotemporal warping. The

trilinear interpolation is parametrized by a flow estimation network. The spatiotemporal flow estimation network takes a fixed number of frames as input (H × W × C × T) and does not perform temporal convolution. Instead, the temporal dimension is merged with the channels, with filters of size (3 × 3 × (C1 × T) × C2). The produced spatiotemporal flow is a spatial map with 3 flow channels (H × W × 3), representing the spatiotemporal flow for the center (reference) frame only. This architecture is designed for the very specific task of video restoration, working on small temporal neighborhoods.

To simplify notations, we use the name "STN" to refer to transformation networks, and specify the dimension when needed. This also avoids confusion between Spatial Transformer Networks and the Transformer architecture of Vaswani *et al.* (2017). Another way to implement a spatiotemporal transformer network, with crop, translation and scaling in three dimensions, is to directly extend the original idea of Jaderberg *et al.* (2015), using affine transformations defined with few parameters. This can easily be adapted to 3D data, with affine transformations defined by 3 × 4 parameters that can be constrained further. However, with spatiotemporal data, the issue of varying video duration arises. To obtain a fixed number of parameters from an input of arbitrary duration, we can apply adaptive pooling on the input data volume of the STN parameter network. Length uniformization can also be applied to the actual volume to which the transformation is applied. The mechanism works on relative coordinates, so we can consider it is robust to size variations.

The issue with spatiotemporal transformation network is that of short-clip training. It is impossible to model video-level attention when training only on short clips of 16 frames. Although the parametric model has the theoretical capability to adapt to different input sizes, through its relative coordinate system, the STN doesn't have the opportunity to develop the relevant behavior within short clips. While a STN could focus on salient frames or avoid occluded ones in a long video, this becomes difficult to do within a small clip, as all frames tend to be similar. Thus, during training, the STN can not be effective, and doesn't learn to become effective for the inference phase with long videos.

4.4 Self-attention

Self-attention was particularly well described in the work of Vaswani *et al.* (2017), in which the Transformer architecture was developed. This attention mechanism consists in weighted averages of feature values, producing a new representation of the input. Attention weights are computed from pairwise interactions of different positions within a single feature map, or sequence. Such alignment of entities within the same set is called intra-attention or self-attention.

4.4.1 Inter-attention

The idea of intra-attention is developed in opposition to the mechanism of inter-attention, also called inter-alignement, notably discussed in Bahdanau *et al.* (2015) for NMT. With inter-attention, target words from the output sequence are aligned to source words from the input sequence. The model computes an alignment probability $\alpha_{i,j}$ to determine the importance of the feature vector h_j, representing the source word x_j, in the generation of the target word y_i. Indeed, the generation of y_i is based on a context vector, or attention vector, c_i, with:

$$c_i = \sum_{j=1}^{L_x} \alpha_{i,j} h_j. \tag{4.3}$$

This alignment mechanism implements an attentional behavior, with the model deciding which parts of the input to focus on at each step of the generation process.

The vector c_i presented in Equation 4.3 is a contextualized attentional representation of the input features, or weighted feature vector. Note that while the names "context vector" and "attention vector" are widely used in literature, they are misleading and not well defined. We consider the attention vector refers to the vector of attention weights α, and the context vector more specifically refers to a representation of the context information that is used to produce the attention vector, similar to the idea of query as we will see.

Xu *et al.* (2015) adapt this soft attention weighting to the task of image captioning, replacing the source sentence with a source image. Choi, Cho & Bengio (2018) also weight the feature dimensions (channels) of the context vector to produce the new representation.

Inter-alignment seems to be limited to frameworks of generation tasks, e.g. producing sequences. An attention vector is computed for every step with the interaction of two sequences: each consecutive state of the generation network (which can be seen as the query) is aligned to every position from the input sequence. In contrast, intra-attention can be applied to classification tasks.

4.4.2 Self-attention

In self-attention, alignment is computed between every positions from the same sequence. Equation 4.3 still applies. However, $\alpha_{i,j}$ measures the interaction of h_i with h_j. Also, h_i and h_j come from the same source sequence, and they are respectively considered as a query and a key-value pair.

To make the compatibility measure more fruitful, the input is first transformed. Linear projections are used to map each feature vector h_j to three different embeddings: query, key and value (Q, K and V). The vector c_i, is the new representation of h_i, computed as a weighted average of all values.

$$c_i = \sum_{j=1}^{L_x} \alpha_{i,j} V_j. \tag{4.4}$$

The weights $\alpha_{i,j}$ assigned to each value V_j are determined by the compatibility of the corresponding key K_j with a query Q_i. The compatibility function measures the interaction of K_j with Q_i. It determines whether V_j is relevant for representing the feature h_i. The dot-product operation is illustrated in Fig. 4.2. Softmax is applied on the attention profile α_i, and the result of the dot-product of α_i and V is aggregated by summation of its entities, as in a weighted pooling mechanism.

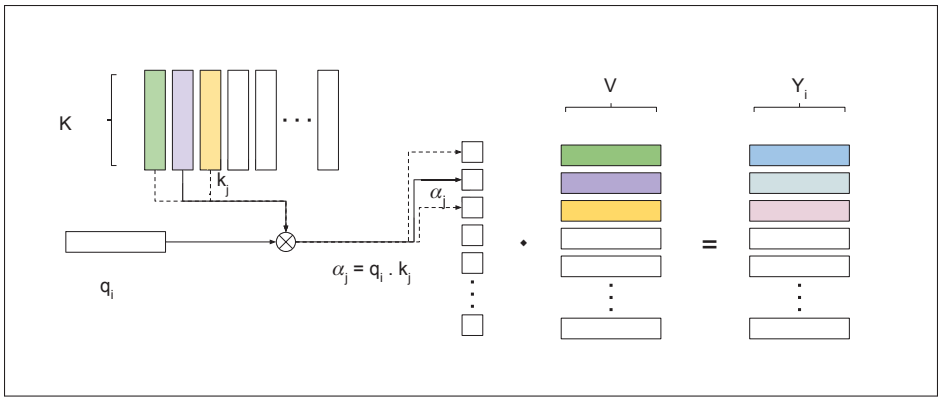

Figure 4.2 Dot-product operation between query, keys and values, to build a new representation with self-attention

• **Compatibility function:** Two compatibility functions are usually considered: additive attention, and multiplicative (dot-product) attention. Bahdanau *et al.* (2015) use additive attention in an alignment model which scores how well the inputs around position j and the output at position i match:

$$\alpha_{i,j} = \sigma(W * tanh(W_1 * s_{i-1} + W_2 * h_j)) = \sigma(W * tanh(q_i + k_j)). \tag{4.5}$$

With W the weight matrix of a fully connected layer. The softmax activation σ ensures the sum of attention weights over all positions is one. Additive attention typically computes the compatibility function using a feed-forward network with a single hidden layer. This approach is considered similar but less efficient than dot-product attention (Britz *et al.*, 2017).

Vaswani *et al.* (2017) use dot-product attention. The product is scaled by a factor $1/\sqrt{d_k}$, with d_k the feature depth of the attention embeddings. This is supposed to limit the growth of the dot-product resulting values, which can cause issues in the softmax gradients.

$$\alpha_{i,j} = \sigma(\frac{q_i \cdot k_j}{\sqrt{d_k}}). \tag{4.6}$$

This computation can be performed efficiently with matrix multiplications, by stacking several queries, keys and values into matrices to process them in parallel. An overview of this matrix computation is presented in Fig. 4.3.

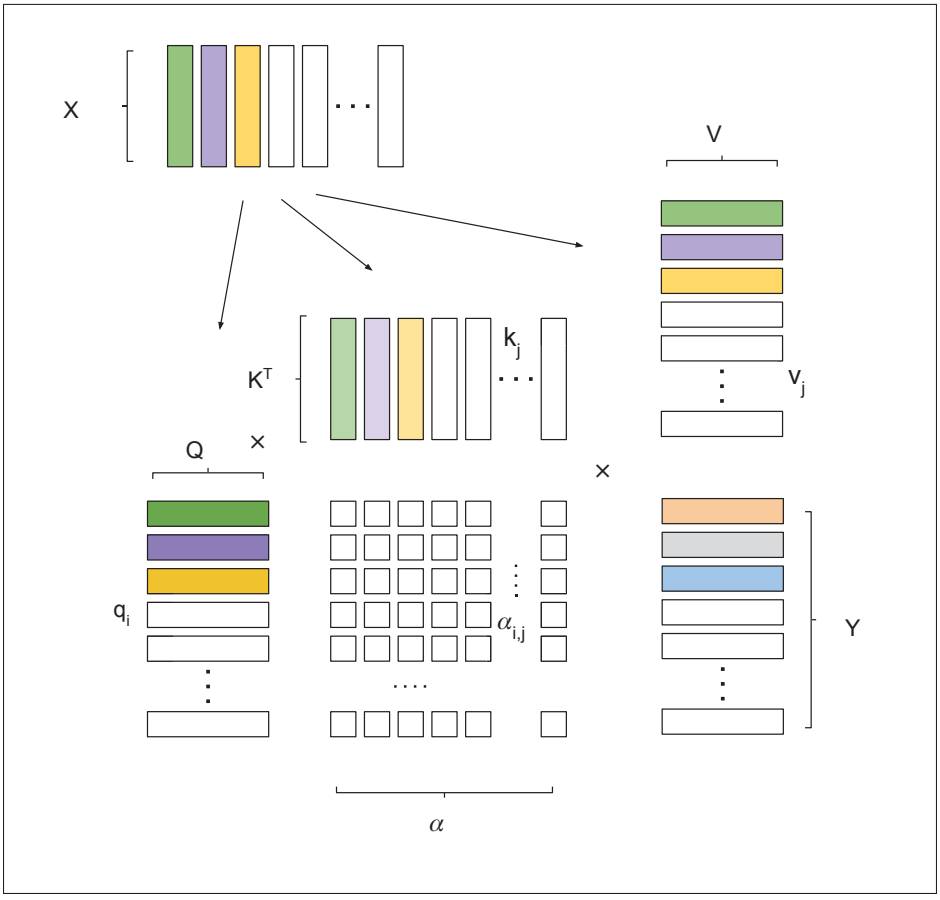

Figure 4.3 Evaluation of several query-key interactions in parallel with matrix computation

$$C = \sigma(\frac{QK^T}{\sqrt{d_k}})V. \tag{4.7}$$

• **Dot-product self-attention:** Self-attention is sometimes confused with the concept of dot-product attention, as opposed to additive attention. However, we consider self-attention refers to a mechanism which uses a reflective compatibility function, between each pair of entities within the same source embedding. Output entities can contain information from other entities, while

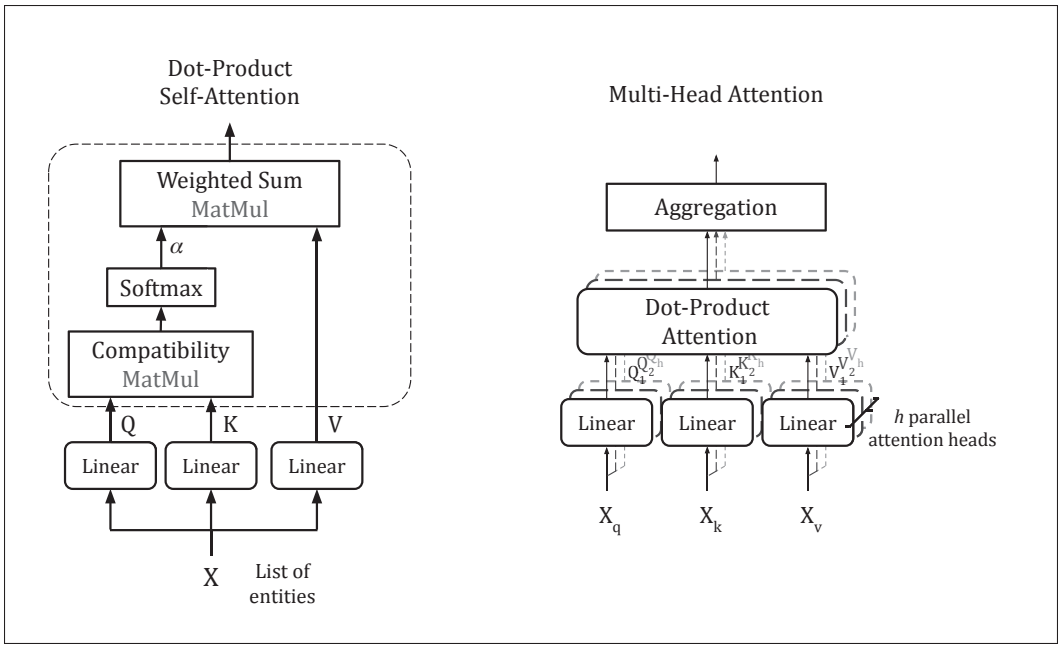

Figure 4.4 The architecture of a multi-head dot-product attention module, developed in the original work on the Transformer, Vaswani *et al.* (2017). For self-attention, all queries X_q, keys X_k and values X_v come from the same sequence, volume or set of entities. Computations for all queries are parallelized with matrix multiplication (MatMul)

mask attention is only a weighting or gating mechanism. For a representation produced with self-attention, all positions in the source sequence are being aligned with all the positions in this same source sequence. The resulting $L \times L$ attention map can be produced with dot product or a fully connected layer (as in additive attention).

Indeed, dot-product attention doesn't refer to the operation that realizes the weighted sum of entities with an attention profile, but to the operation that produces the attention map from features and context/query vector (with the idea of measuring compatibility between these elements). If attention weights are not computed for each pair of entities, but only between each entities and a unique query or context vector, this does not correspond to self-attention, but a case of mask attention, with the attention mask taking context information into account. And this mask can be computed with dot product or additive attention.

The popular work of Vaswani *et al.* (2017) efficiently implemented self-attention with a dot-product operation, which lead to some confusion in other work. The proposed Transformer architecture was designed to improve computational efficiency and parallelization within examples, by avoiding the sequential computation along the symbol positions of the input and output sequences. Stacked self-attention modules organized in an encoder-decoder scheme completely replace convolutional or recurrent layers.

• **Multiplicative interactions:** "Attention mechanisms have become an integral part of compelling sequence modeling and transduction models in various tasks, allowing modeling of dependencies without regard to their distance in the input or output sequences" (Vaswani *et al.*, 2017). This has advantages reaching beyond the attentive behavior, as explained in Wang *et al.* (2018b). The self-attention mechanism is used as a feature extraction module, able to draw global dependencies (as opposed to convolutional layers, with limited receptive field).

Wang, Li, Li & Van Gool (2018a) also use multiplicative interactions as the basic component of an appearance-independent relation detector. A discussion on multiplicative interactions can be found in Memisevic (2013). The advantage of multiplicative interactions is their ability to detect transformations between frames, while being independent to the content (contrary to convolutions). A learned gating mechanism on the outer product of the two frame x and y representations can identify transformations:

$$z_k = \sum_{i,j} w_{i,j,k} x_i y_j. \tag{4.8}$$

Here, the transformation code z represents the linear transformation g relating both frames, that is $y = g(x)$. A weight tensor w operates a linear transformation on the dot-product.

They show that a gating of the outer product of two vectors (representing flattened frames) is approximated (with a factorization) by applying linear transformations on the vectors before computing their dot-product. The factorization actually applies constraints on the weights that facilitates training, with fewer parameters.

$$z_k = \sum_{i,j}(\sum_f w^x_{if} w^y_{jf} w^z_{kf}) x_i y_j = \sum_f w^z_{kf}(\sum_i w^x_{if} x_i)(\sum_j w^y_{jf} y_j). \tag{4.9}$$

Memisevic (2011) provides more thoughts about the role of multiplicative interactions, and makes links with energy models: "An energy model is a computational unit that relates images by summing over squared responses of, typically two, linear projections of input data. This operation can be shown to encode translations independently of content".

This ability of multiplicative interactions to learn and infer relationships in data leads Wang *et al.* (2018b) to use the mechanism of self-attention as the principle spatial feature-extraction component of their model, rejecting the idea of attentional behavior. On the contrary, Zhu, Cheng, Zhang, Lin & Dai (2019) experiment with self-attention in images, and conclude that multiplicative interactions are not particularly relevant for self-attention.

• **Attention queries:** One attention profile (a set of k values α, that sum to 1, defining attention over all keys) is produced for each query independently. QK^TV results in a new representation with q entities (i.e. the number of queries). Typically, we consider an input x that will be transformed to Q, K and V. The number of queries and the number of keys is identical. Thus the output is a new representation with the same size as x. This can be seen in the work of Vaswani *et al.* (2017); Zambaldi *et al.* (2019); Wang *et al.* (2018b). For classification tasks, we can produce a set/sequence of representations with different queries coming from all positions, before aggregating this set. Alternatively, we can consider having only one query which represents the entire video. This would result in a weighted pooling mechanism, producing a single output from a sequence. Fig. 4.5 compares both approaches.

• **Multi-head attention:** Attention heads allow to model simultaneously several types of dependencies, or semantic rules, in the data. Several scaled dot-product attention modules are used in parallel, with different learnt embeddings Q, K and V. One attention profile is produced for each query-head pair independently. Outputs of all attention-heads can be summed or concatenated or fed to a fully connected layer.

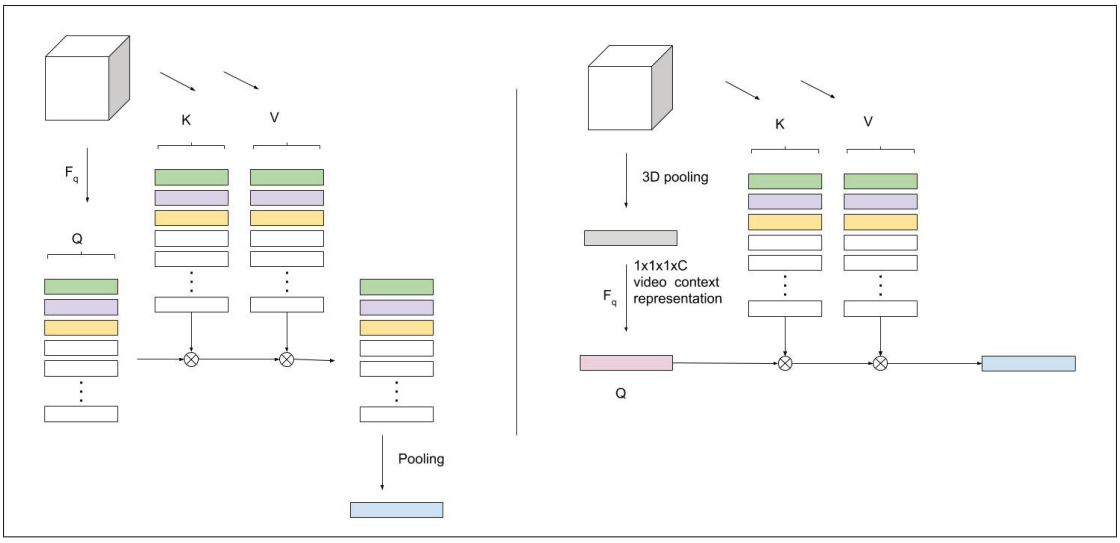

Figure 4.5 Left: performing self-attention with interactions of each position pairs. Right: using a single query for the entire video

• **Residual Connections:** The output of a self-attention layer is often summed with the input in a residual connection. This is particularly important when placing a self-attention module within a pretrained model, as the outputs are changed by the value layer, which is initialized randomly.

• **Positional encoding:** Sequence or spatiotemporal volume is transformed to a set of entities that are processed independently of their order and position. In some cases, the compatibility function might benefit from this information, for instance to find relationships between entities based on their relative positions, or their distance. To make this possible, position can be encoded in the entity representation (mostly q and k) in order to compute the interactions. For the Transformer architecture, Parmar *et al.* (2018) and Vaswani *et al.* (2017) encode the positions with learned parameters, or sine and cosine functions that are summed to the feature maps. Zambaldi *et al.* (2019) add three channels to the feature dimension to include x, y and t positions. Shaw, Uszkoreit & Vaswani (2018) propose a relation-aware self-attention, their Transformer modulates attentions based on pairwise distance instead of absolute positions. The compatibility function is the sum of two terms: content compatibility and position compatibility. The use of a simple addition operation to incorporate relative positions is justified by computational requirements. The efficient parallel computation of self-attention from Vaswani *et al.* (2017),

with batches, heads, queries and keys, through matrix multiplication is possible because keys are the same for all queries in the sequence. This prevents from integrating information of entities distance directly in the query-key multiplication. The Music Transformer (Huang, Vaswani, Uszkoreit, Simon, Hawthorne, Shazeer, Dai, Hoffman, Dinculescu & Eck, 2019) improves the mechanism with a memory efficient implementation of relative positional encoding, to manage longer sequences for generative modeling of symbolic music. Other studies (Wang *et al.*, 2018b; Dai, Das, Minciullo, Garattoni, Francesca & Bremond, 2021) do not consider positional encoding to be beneficial and do not implement it. Notably, positional encoding seems more important for generative tasks (e.g., sequence to sequence, image to image) than classification.

4.4.3 Spatio-temporal self-attention

For a Transformer architecture adapted to image generation, Parmar *et al.* (2018) use local self-attention, with a small spatial receptive field. The module only computes relationships between neighboring pixels, to reduce computational complexity.

For the task of multidocument summarization, generating Wikipedia articles, Liu, Saleh, Pot, Goodrich, Sepassi, Kaiser & Shazeer (2018c) also used local attention inside small blocks of text, coupled with coarse global attention (with downsampled keys and values using strided convolutions).

On the contrary, Wang *et al.* (2018b) studies spatial self-attention as non-local operator. The authors insist on the idea of capturing non-local dependencies, more than developing an attentional behavior. Indeed, they remove the softmax operation on the weights (which ensures a total attention score of 1). This study considers self-attention as a special case of non-local filtering. To reduce the computational cost of self-attention over all pixels, they propose a sub-sampling of K and V.

Similarly, Zambaldi *et al.* (2019) use the self-attention mechanism but consider it as a relational block, carrying out feature extraction, as an alternative to convolutional layers. The feature map is considered as a set of entities (each location in the feature map is seen as an entity).

This is simply achieved by flattening the tensor, and can be used with spatiotemporal data as shown in Fig. 4.6. Self-attention is used to compute interactions (compatibility) between each pair of entities. With this setup, the relational block performs non-local pairwise relational computations, complementing the local-feature extraction of convolutional layers as in Wang *et al.* (2018b). They also propose to stack multiple relational blocks to allow for higher-order relational computations. Stacking several relational blocks with shared parameters introduces a recurrence computation mechanism.

Spatial and temporal non-local operators can also be separated, as Tran *et al.* (2018) alternated spatial and temporal convolution in the R(2+1)D architecture. For instance, Aksan, Cao, Kaufmann & Hilliges (2020) propose a decoupled temporal and spatial attention for spatiotemporal, 3D human motion generation. “For every joint we define temporal attention over the same joint in the past and spatial attention over the other joints at the same time step” (Aksan *et al.*, 2020).

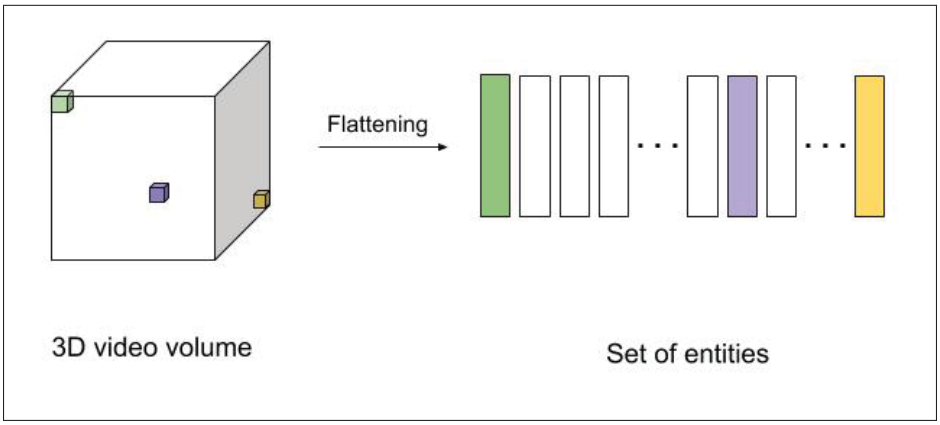

Figure 4.6 Spatiotemporal feature maps viewed as a set of entities to compute non-local relations (Zambaldi *et al.*, 2019)

With the Video Action Transformer, Girdhar *et al.* (2019) revisits the idea of self-attention, by only using specific entities as queries, instead of using them all. In the context of action recognition, features associated to region proposals act as queries. Keys and values come from all the entire clip feature volume. These features are weighted based on their interactions with the query person (a block extracted around the subject with ROI-pooling). “The intuition is that the self-attention will add context from other people and objects in the clip to the query

vector, to aid with the subsequent classification" (Girdhar *et al.*, 2019). The authors consider the keys to encode the context information. This shows that the notion of context is not well defined, and how unclear the specific role of each component is in this deep learning mechanism. However, the computation remains the same, and only its interpretation differs. We consider that features, through their key and value mappings, are weighted in light of context information provided by the query. Different queries provide different contextualization, with different feature compatibility, resulting in different attention profiles.

4.5 Results and discussion

The experimental methodology is similar to Chapter 3. Attention mechanisms are evaluated on the AFEW dataset for emotion recognition, and complementary results are provided on HMDB-51 for action recognition (see Section 3.3.1). Attention modules are implemented in the inflated 3D VGG-16 model and the Inception-v1 I3D. The baseline is provided by the models without attention mechanisms. An ablation study is proposed to characterize parameters and attention dimensionality for each type of attention mechanism. Results obtained with the different methods are then compared and discussed.

4.5.1 Mask attention

The attention module is placed between the (pre-trained) feature-extraction layers and the classifier, as in Aminbeidokhti *et al.* (2019). In our preliminary experiments, placing mask attention before the pretrained CNN, directly working on input pixel didn't produce any relevant behavior. When placed after the CNN, the attention layers benefit from the rich features extracted by the CNN, enabling more complex behavior. The downside is that map resolution is very low at the end (it is a benefit in terms of computations however).

We experiment with two layers of 1x1x1 convolutions, equivalent to fully connected layers operating on each spatiotemporal position independently. We use $tanh$ activation:

$$o_{x,y,t} = 1 + tanh(W_2(W_1 f_{x,y,t} + b_1) + b_2). \tag{4.10}$$

Attention masks presented in Figures 4.7 to 4.10 show how this simple model can mask out background precisely to complement the face-alignment process. The masks can also put more emphasis on eyes or mouth depending on the expression, but several factors limit this behavior: the masks have low resolution and neighbouring spatial positions are highly correlated because of overlapping receptive fields in the CNN producing the input feature maps. Consequently, frame-level masks tend to be similar to their temporal neighbours.

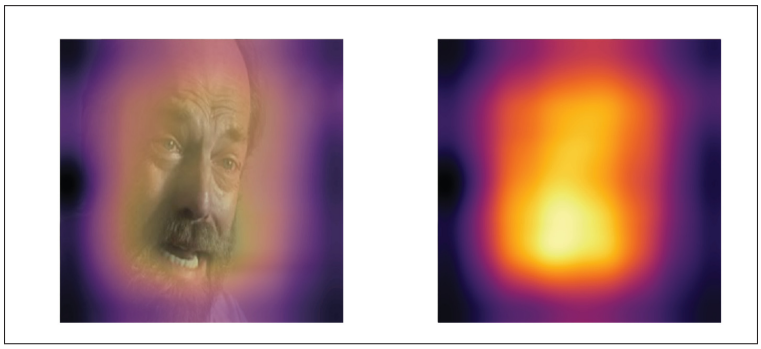

Figure 4.7 Illustration for a single frame of spatiotemporal attention mask for example *005711000* of AFEW

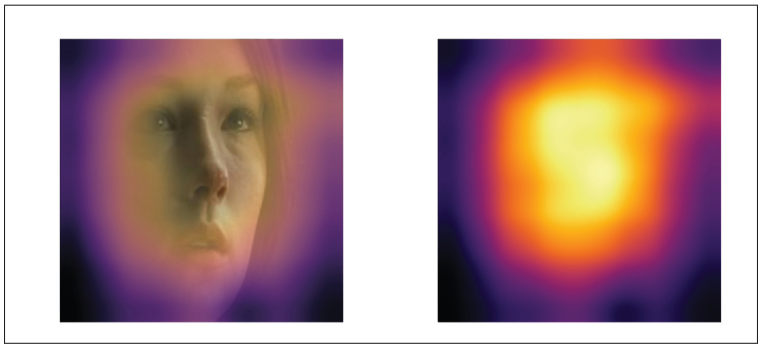

Figure 4.8 Illustration for a single frame of spatiotemporal attention mask for example *005405240* of AFEW

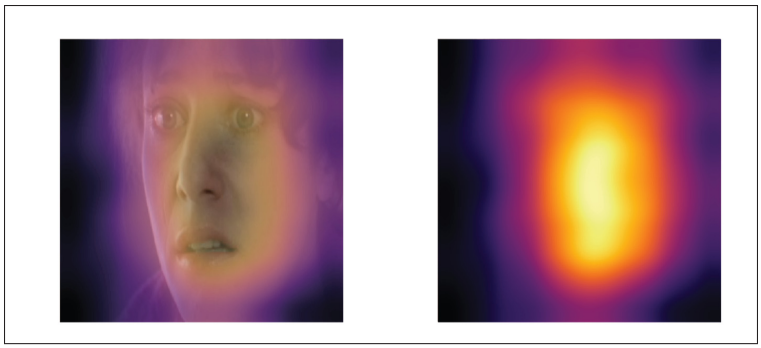

Figure 4.9 Illustration for a single frame of spatiotemporal attention mask for example *004041920* of AFEW

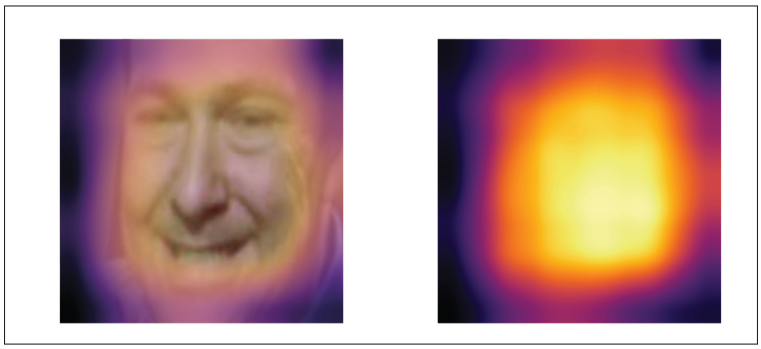

Figure 4.10 Illustration for a single frame of spatiotemporal attention mask for example *001444527* of AFEW

Results with different dimensionalities are reported in Table 4.1. Temporal masks, with a single attention weight per frame, produces uniform attention, which does not provide any benefit. One issue is the limitation of short-clip training. As different frames present within a clip are usually similar, the model cannot learn any relevant intra-clip weighting behavior. If attention is not relevant within short clips, the model cannot learn to perform it in entire videos. The model should be able to learn to mask out occluded frames if the occlusion is temporally very limited withing the clip. This probably does not happen because the mask is regressed from the spatiotemporal features produced by the VGG network, which are already influenced by neighbourhoods of frame, given the temporal receptive field of the stacked convolutions. Because of this, the VGG network might implicitly manage to avoid very short occlusions, without attention.

Table 4.1 Evaluation of different types of mask attention mechanisms (AFEW)

Method	**Accuracy (%)**
Baseline	45.73 ±0.21
Temporal mask	45.64 ±0.24
Spatial mask	45.82 ±0.17
Spatio-Temporal 3D mask	45.94 ±0.18
Context concat. 3D mask	45.53 ±0.28
Context additive 3D mask	45.39 ±0.27
Context mult. 3D mask	46.22 ±0.08

This shows that the spatiotemporal 3D masks mostly benefit from the spatial attention behavior. However, the improvement over the baseline is very limited. The masks can complement the face alignement mechanism by masking out the background more precisely, and also put more emphasis on parts of the face, but the masks remain very coarse.

With our initial mask attention mechanism, each spatiotemporal position is gated through multiplication with a float value (0 masks the features, 1 has no influence, and greater values can highlight the features). The attention weight is produced by a learned transformation of the feature representation of the given spatiotemporal position.

With context-based mask attention, the mechanism still operates a gating of each spatiotemporal position, but the attention weight is based not only on the features at this position, but also on a global representation of the entire video, summarized in a context vector, through a linear transformation, ReLU, and pooling.

Table 4.1 compares different implementations for incorporating context information in the production of the attention mask (based on the discussion on Compatibility functions, in Section 4.4.2). In our experiments, concatenation and additive mechanisms are not able to leverage this additional information, which tends to deteriorate the performance. On the contrary, a multiplicative mechanism, with dot-product in the channel dimensions of the feature and context vector, is able to produce relevant attention weights for each position, conditioned on the

global information of the video. The resulting attentional behavior can be considered similar to what could be obtained by using a self-attention mechanism with only one query per video, with the query summarizing the global video information, similarly to the context vector. Also, this raises questions about the relevance of learned transformations Q, K and V.

The performance of this context-based mask attention shows that the model can learn a relevant attention behaviour within short clips, and generalize it for entire videos during testing.

4.5.2 Transformation networks

Table 4.2 Evaluation of different modalities of STN modules (AFEW)

Method	**Accuracy (%)**
Baseline	45.73 ±0.21
Spatial STN	45.71 ±0.18
Temporal STN	44.82 ±0.26
Spatio-Temporal STN	45.33 ±0.21

Table 4.2 summarizes our experiments with spatial, temporal and spatiotemporal versions of the STN attention module. Frame-level STN does not provide any interesting behavior for the model. Images are already aligned to faces, and the transformation of our STN is too constrained to improve it, as opposed to our spatial mask attention. More complex implementations of STN could be beneficial, e.g. regressing the entire sampling grid (Caballero *et al.*, 2017; Dai *et al.*, 2019). We do not experiment such methods as they are much more computationally complex, and usually reserved to specific tasks. An alternative approach would be to use several attention heads, with dedicated networks working on the different crops. This does not fit our framework and pretrained 3D model, as discussed about the Recurrent Attention Model (Section 4.1.2).

For the spatiotemporal 3D STN, we create a localisation network of three layers to produce a set of 3×4 parameters for a 3D affine transformation of the input spatiotemporal volume. The network computes transformation parameters from a resized, uniformized input of fixed size (spatially and temporally) and produces parameters for a transformation working with relative

positions. We use tanh activation on parameters, with an initialization to identity map. The transformation is applied on the input of arbitrary length.

We also experiment an alternative of the 3D STN that is constrained to only temporal transformation. Parameters related to the spatial dimensions are fixed to operate identity mapping. This 1D transformation only has two free parameters. It operates a temporal cropping of the video input. This mechanism has a negative impact on the model's performance, by introducing noise and reducing the number of frames in training clips which is already very limited. Again, the sampling mechanism is too constrained to avoid unwanted frames, unless they are at the beginning and end of the clips. Even then, the transformation network, which is based on sampling with interpolation, is not adapted to our needs, especially as we work with very few entities in the temporal dimension. Temporal crops and zooms within clips of 16 frames produce bad-quality representations.

4.5.3 Self-attention

In this section we provide visualizations of the behavior of our temporal self-attention mechanism. The temporal attention map has two dimensions, query and key, and shows the compatibility score of each key for each query. Each line can be read as the attention profile of a single query, softmax is applied along each line so the attention weights sum to 1 for each query. These attention scores are used to weight the influence of all frames in the production of a new representation of a given frame (Section 4.4).

To have an overview of the attention at the video level, we average the attention profile of all queries. This gives the relative importance of each frame in the video as a 1D temporal attention profile.

For an important number of videos, we observe that attention maps are dominated by vertical line, suggesting the queries are the same for all the frames of a video (as seen in figures 4.20, 4.12, 4.13 and 4.16). If the queries of all frames are the same, this means we could simplify the self-attention module to produce a single query for the entire video. A single

query would act as a context query, with information about the entire video, rather than having a specific query for each frame to produce a representation of this frame based on a relational behavior with all other frames. The latter behavior might not be relevant to our classification task, while it was designed for a generative sequence-to-sequence task.

Note that if the queries do not capture any information specific to the frame or the video, or if we skip the queries and use the keys directly, the attention mechanism becomes equivalent to our mask attention (Section 4.2).

• **Expression intensity:** The most typical behavior observed with temporal attention is a focus on frames displaying intense expressions, and an inhibition of neutral frames. With self-attention, expression intensity of each frame can be encoded in its key representation, allowing for a temporal distribution of attention based on intensity.

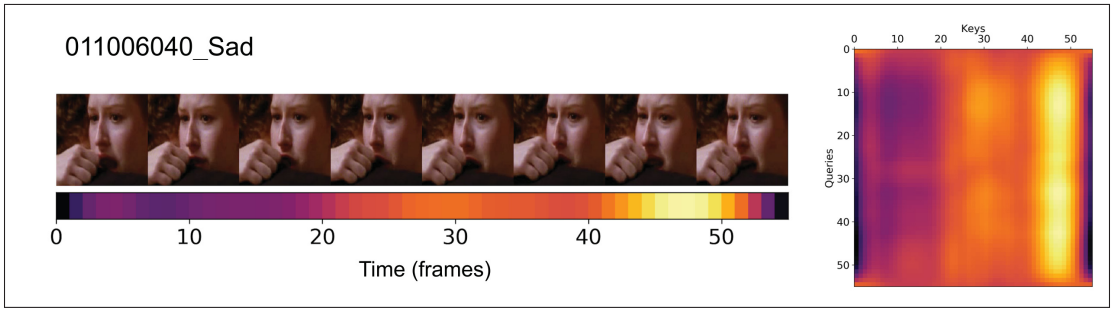

Figure 4.11 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *011006040*

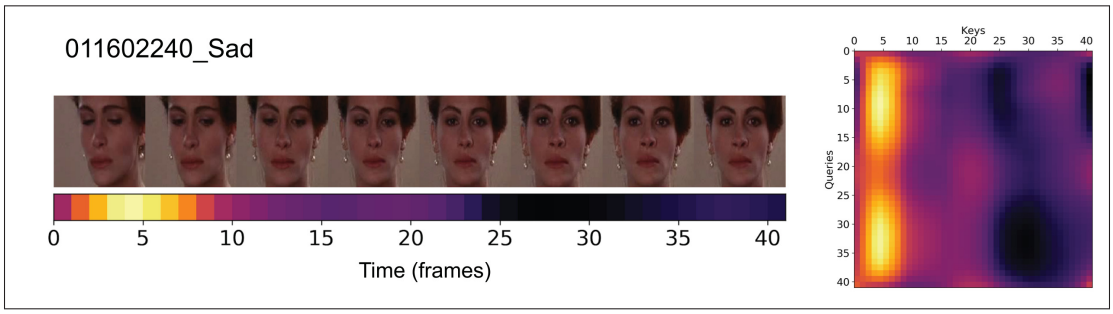

Figure 4.12 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *011602240*

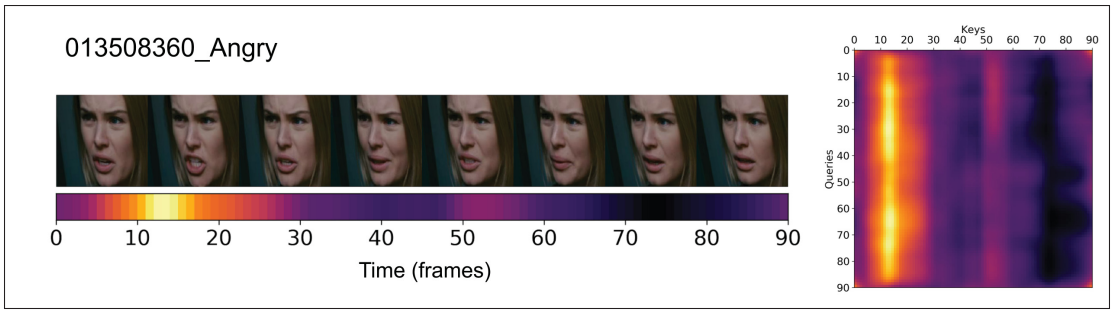

Figure 4.13 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *013508360*

For Figures 4.11 and 4.12 respectively, the expression of sadness intensifies and decreases through the video but remains visible at all time. While changing in intensity, the expression remains consistent, there are no important variation in head-pose, lighting conditions, so all frames produce similar queries that favor the most expressive ones.

Similarly with the example of Figure 4.13, the expression of anger fades but remains visible with the frowning notably. All frames produce similar queries that focus on the most intense expression of anger.

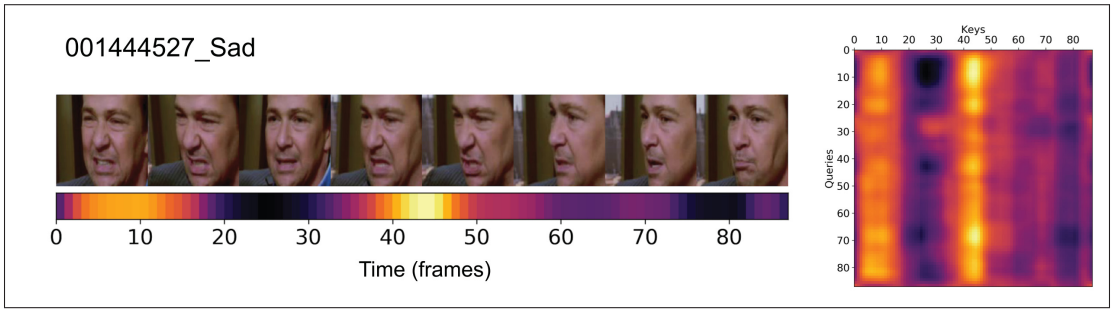

Figure 4.14 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *001444527*

Figure 4.14 presents a more complex attention profile due to variations of expression intensity. We note that the correct label is sad, but the attention is focused on frames that might contain more ambiguous expression, between anger, disgust and sadness, but at least avoids neutral frames.

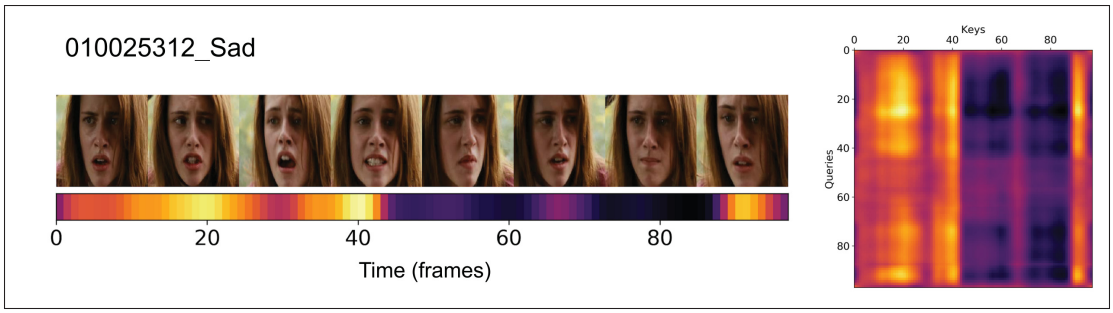

Figure 4.15 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *010025312*

With Figure 4.15 we see a similar behavior, with attention being focused on strong expressions of sadness. The more neutral frames of the middle are not being focused on. Even more interestingly, on the side of queries, these less expressive frames do not produce discriminant queries, resulting in horizontal lines of relatively uniform attention in the center of the attention map. On the contrary, the frames in the beginning and end of the video show intense expression of sadness and are being related to each other, and not to the neutral ones.

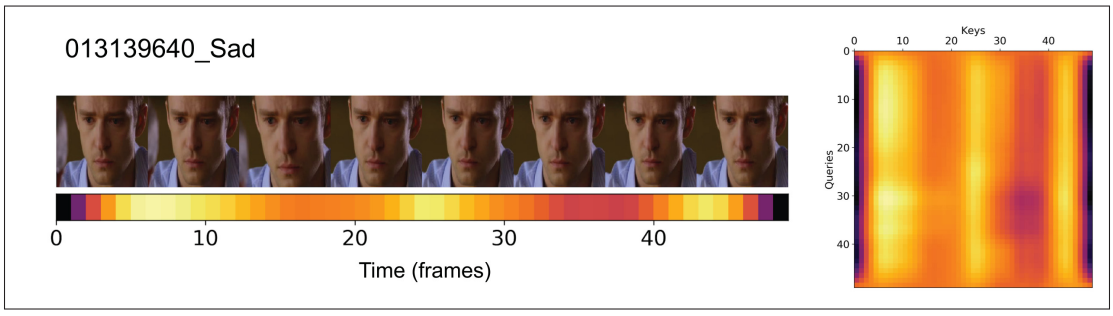

Figure 4.16 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *013139640*

These attention profile can be compared to Figure 4.16, that displays a video example in which all frames look identical, so the attention is practically evenly distributed.

• **Expression disambiguation:** Beyond focusing on most expressive frames, the query mechanism of self-attention implements relational computations that allows for disambiguation of certain frames, by interpreting them in light of the rest of the video. Queries can be seen as

encoding this context information, and we verify this behavior when visualizing the temporal attention profile obtained by averaging over queries.

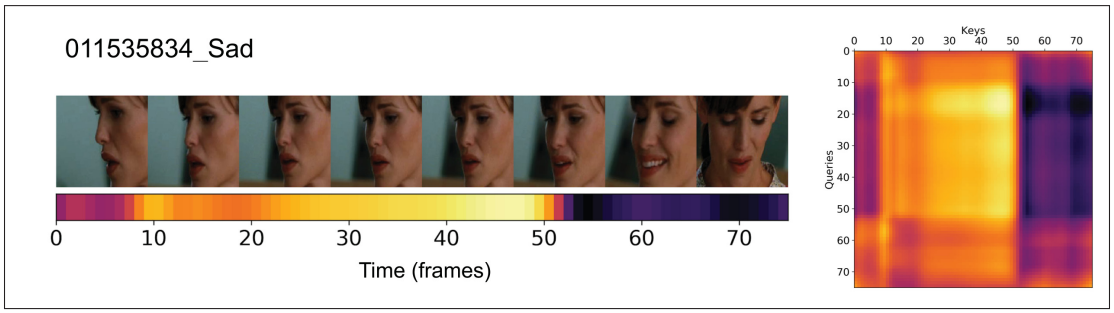

Figure 4.17 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *011535834*

Figure 4.17 shows an interesting behavior, as the ambiguous smiling expression at the end has to be understood in light of the consistent sadness expression that precedes. In the attention map, the vertical separation between bright and dark areas, around frame 50, show a clear difference in key representations between the consistent sadness expression (frames 10 to 50) and the following ambiguous frames. For the queries, this difference translates into lines with high contrast, sharp attention, and lines with low contrast, blurred, distributed attention. Indeed, the queries corresponding to ambiguous frames do not encode precise context, while the clearly sad expression frames provide rich queries that favour similar sad expressions, and are not compatible with the smile.

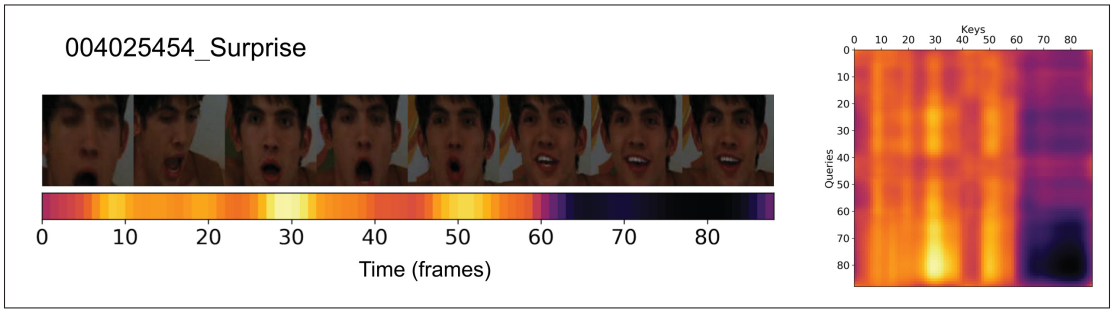

Figure 4.18 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *004025454*

Also, the end frames of Figure 4.18 are ambiguous, with a big smile but eyes wide open indicating surprise. The corresponding queries focus attention on the frames showing a more canonical expression of surprise, removing the ambiguity in the representation.

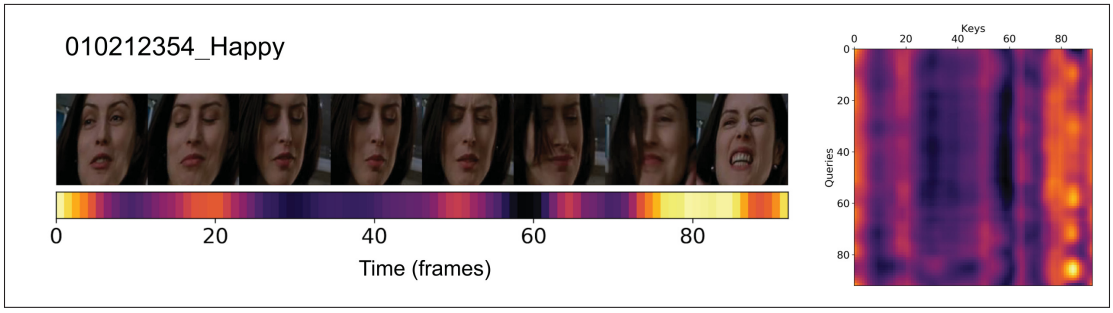

Figure 4.19 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *010212354*

In Figure 4.19, center frames display a shy smile, but the face is oriented downward and frowning. Again, the self-attention mechanism removes ambiguity by focusing attention on the more expressive frames.

• **Capture noise:** Issues of head-pose variations, lighting conditions, or imprecise temporal delimitation of expression videos can lead to very diverse types of noise in the emotion video. Keys provide a first way to reduce noise, by inhibiting frames that do not display emotion. When coupled with the mechanism of queries for self-attention, even more complex behaviors can emerge.

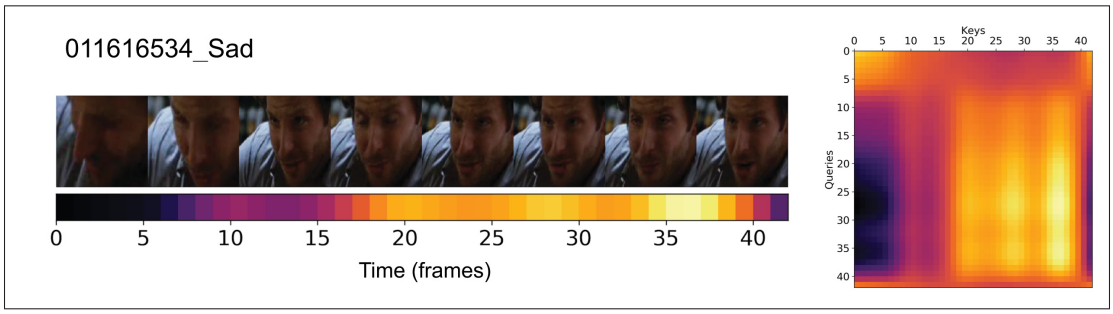

Figure 4.20 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *011616534*

In the example of Figure 4.20, the first frames have pose issues and the facial expression is not clearly identified. Consequently, these frames do not receive attention. More interestingly, the attention map shows the first frames produce non-discriminant queries (as they do not contain relevant context information) with distributed attention, but the frames showing clear expressions of sadness produce discriminant queries that favour similar expressions, with intense attention.

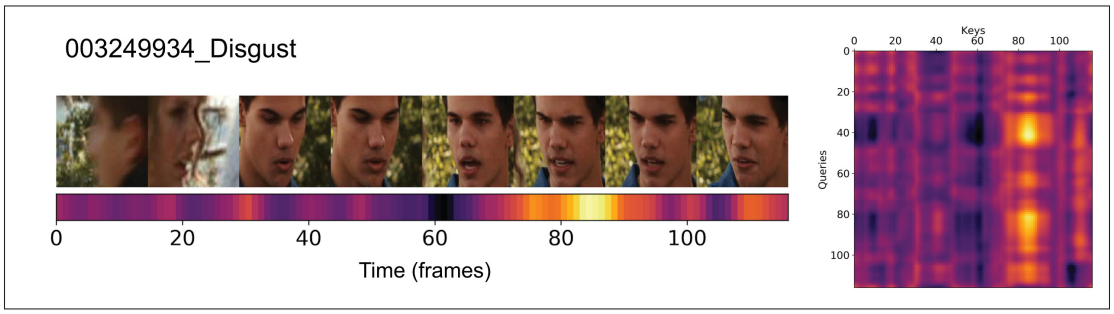

Figure 4.21 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *003249934*

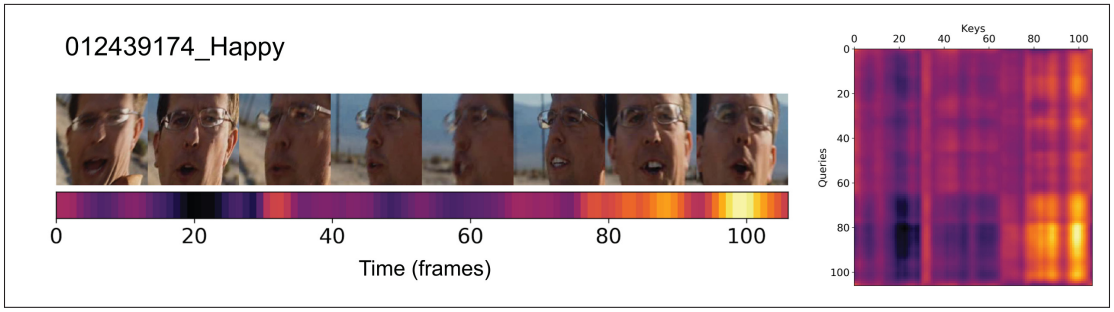

Figure 4.22 Visualization of query-key compatibility matrix and corresponding temporal attention profile for AFEW video *012439174*

Figures 4.21 and 4.22 have very complex attention maps because of the poor temporal cropping, with several actors in the scene, and variations in lighting and pose, as well as inherent evolution of the face expression. This results in very different queries being produced for each frame. Consequently, an intense expression displayed by a person that is not the main subject of the video does not receive much attention, as it is not compatible with most queries. The resulting temporal attention profile, obtained by averaging over queries, shows a relevant attentional behavior, that favors the most expressive, less noisy and less ambiguous frames.

The self-attention mechanism is able to extrapolate attention modelling from short-clip training to full-video testing. The operation is not only attentional behavior, but also helps capture temporal relations and features. The temporal self-attention mechanism is placed at the end of our pretrained VGG model, which avoid interfering with the flow of information, representations and channels activations. Contrary to temporal stochastic softmax (Chapter 3), self-attention weights do not depend on target class. Instead, attention depends on queries, which provides a context information, which can be richer than the categorical representation, but has to be extracted from data, without explicit supervision.

Self-attention would operate a weighted pooling behavior if we were using a single query for the entire video, that can provide information about general context, like the range of expressions, average pose, subject features, to then evaluate each frame in regards to the general video information (and for instance identify irrelevant frames, that do not show the main actor).

This spatial attention operates on the frame level. While in the temporal experiment, we model attention over a sequence of frame representations (spatially pooled, and passed through the fully connected layer of the VGG network). So the queries and values are much more local and not as descriptive.

This is a limitation of spatiotemporal attention, partly due to the general implementation (from video capture to architecture design), but also inherent to the task. Averaging spatial positions to obtain a sequence of frame representations works well. On the other hand, averaging temporally over frames, to obtain a representation of each spatial position is not directly possible because of inconsistencies in the alignment process, different scenes in the frame sequence, and head-pose variations that cannot be corrected by the system. Even with a perfect face alignment, it is clear that a frame (temporal slice) is more informative than a spatial slice (temporal vector of only one spatial position). And the reason might not only be that we compare a 1d and a 2d data sample. For evolutionary reasons, the face and its expressions are best seen in space.

Self-attention has been shown to work well on spatial data (Parmar *et al.*, 2018; Parmar, Ramachandran, Vaswani, Bello, Levskaya & Shlens, 2019). However, the task and architecture

are quite different from ours. Also, several mechanisms have been used to adapt the Transformer architecture for image data. While positional encoding didn't reveal beneficial for temporal self-attention, it would have probably help with spatial attention. This is suggested by the work of Zhu *et al.* (2019). They actually defend the idea that query content isn't relevant in the Transformer architecture for spatial data. Instead, their results show the efficiency of position-based attention weights. They propose other attention mechanisms that can implement this behavior more efficiently than the self-attention of the Transformer, for instance deformable convolutions as in Dai, Qi, Xiong, Li, Zhang, Hu & Wei (2017).

Spatiotemporal 3D self-attention has the same issue than with spatial attention, features are relatively local. In addition, using softmax in 3D, over an important number of spatiotemporal entities (7x 7x 16) might not facilitate the propagation of gradients for training.

Again, this shows a limitation of spatiotemporal soft-attention. For instance, Li *et al.* (2018a) preferred to use two-step attention, first applying spatial and then temporal attention.

Table 4.3 Study of the number of attention heads and size of attention dimension for self-attention

Configuration		$h \times d_{att}$	**Acc. (%)**
$h = 1$	$d_{att} = 64$	64	45.94 ±0.46
$h = 1$	$d_{att} = 256$	256	46.16 ±0.29
$h = 4$	$d_{att} = 16$	64	45.93 ±0.25
$h = 4$	$d_{att} = 32$	128	46.55 ±0.15
$h = 4$	$d_{att} = 64$	256	46.61 ±0.21
$h = 4$	$d_{att} = 128$	512	46.71 ±0.17
$h = 4$	$d_{att} = 1024$	4096	**46.83** ±0.31
$h = 8$	$d_{att} = 32$	256	45.92 ±0.29
$h = 16$	$d_{att} = 16$	256	45.47 ±0.28

Results presented in Table 4.3 provide characterization of our temporal self-attention mechanism. We study the influence of hyper-parameters of the module, namely h the number of attention heads and d the size of the attention representation in the channel dimension.

We see that increasing channel dimension is beneficial but soon saturates. It also increases the computational cost, the number of FLOPS is $O(n^2 \cdot d_{att} \cdot h)$, with n the number of entities. We can find a good compromise with a dimension of 64 channels for the attention representation (queries and keys). Vaswani *et al.* (2017) use a relation between the number of heads and attention dimension as $d_att = d/h$, to keep the total dimension d of the attention representation and the computational cost constant while increasing head number. However, we observe that performance benefits from the total dimension $h \times d_{att}$, and reducing the size d_{att} of the head representation in order to add more heads is not particularly relevant. On the contrary, having numerous attention heads with too limited representation depth removes the benefits.

We also experimented with a penalization term to favour diversity in the behavior of attention heads as in Lin *et al.* (2017), but this did not improve the performance or change the optimal number of heads.

4.5.4 Comparison

Table 4.4 Evaluation of different types and modalities of attention mechanisms (AFEW)

Method	**Accuracy (%)**
Baseline	45.73 ±0.21
Temporal mask	45.64 ±0.24
Spatio-Temporal 3D mask	45.94 ±0.18
Context mult. 3D mask	46.22 ±0.27
Spatial self-attention	45.95 ±0.28
Temporal self-attention	46.61 ±0.21
Spatio-Temporal self-attention	45.53 ±0.26
Spatial STN	45.71 ±0.18
Temporal STN	44.82 ±0.26
Spatio-Temporal STN	45.33 ±0.21

Table 4.4 regroups results obtained with different types of attention. The mechanisms incorporating context of the entire video (i.e. multiplicative context 3D mask and temporal self-attention)

are the best performers. This is a key component of attention, as entities need to be enhanced or inhibited conditioned on the presence of other features. Results show that this behavior is best implemented with multiplicative operations. This is coherent with observations from neuroscience and biology (Zhou, Xu, Simpson & Cai, 2007; Gabbiani, Krapp, Koch & Laurent, 2002).

Spatial attention provides no significant benefit. The pretrain model is trained to operate on entire face images, with a global approach, as opposed to part-base systems that can be developed with STNs.

Temporal attention on the other hand can be leveraged with our model, but the 16-frame training clips are very limiting. With a relatively complex mechanism as self-attention, we are able to increase performance over the baseline significantly. Again, it is not easy to know the real behavior implemented with this mechanism, between relational and attentional behaviors.

Table 4.5 Validation of temporal attention models on Action Recognition (HMDB-51)

Method	**Accuracy (%)**
Baseline	66.81 ±0.36
Mask attention	66.92 ±0.38
Self-attention	67.25 ±0.35

We further study the impact of attention models for action recognition with HMDB-51. Table 4.5 confirms that a simple temporal mask is not able to provide interesting improvement. Action recognition should benefit from focusing on most informative frames, while inhibiting occluded or irrelevant ones. Temporal self-attention allows a slight increase in accuracy, but remains very limited.

Attention mechanisms have been the focus of a lot of research recently, in the scope of deep learning in general, including FER. Mechanisms labeled as attention are very diverse in terms of behavior, so we tried to clarify the meaning, capabilities and limitations of attention mechanisms for spatiotemporal FER. We limited our study to attention mechanisms that can be incorporated

into existing 3D-CNN architectures, and did not experiment with those designed to replace the entire feature extraction pipeline. These various attention mechanisms have different behaviors, based on very different mathematical operations. Typically, self-attention doesn't only provide attention, it also implements a relational behavior close to feature extraction, and the attention profiles have to be understood in this light. We showed the relevance of context for attentional behavior, which is not always explored in literature.

4.5.5 Directions for video-level temporal attention

Recent studies propose a different approach to model video-level temporal relations, outside the scope of short clips, by working with high-level representations obtained with a frozen feature-extractor. In this approach, the feature-extractor and the attention model are not trained at the same time. This makes it possible to work with more temporal segments at the same time, with limited memory requirements as the feature-extractor is not being trained (gradients are not stored). The feature extractor still processes a video as a set of independent segments (clips of 16 frames), but the high-level temporal attention model can be trained to capture dependencies between segments, that is across the entire video.

However, before training the temporal model, the feature-extractor (e.g. a frame-based 2D CNN or a 3D CNN working on clips) has to be available. This can be an issue on tasks other than action recognition, for which feature-extractors are not readily available. For FER, this imposes a two-stage training process, as we first have to train the feature extractor (or fine-tune a pretrained model) and then train the temporal model with obtained representations.

In the work of Dai *et al.* (2021), the attention mechanism PDAN is a pyramidal implementation of self-attention, but other temporal models can be used in the same way.

Results are reported in Table 4.6. Even without extensive hyperparameter search, we easily obtain improvement over the clip-sampling strategy. The inflated VGG model cannot fully benefit from training clips longer than 32 frames due to its limited receptive field. But the

Table 4.6 Video-level temporal self-attention with two-step training (AFEW)

Range (frames)	**Accuracy (%)**
16	46.61 ±0.21
64	46.53 ±0.10
128	47.14 ±0.19
256	47.39 ±0.22

self-attention module is able to model temporal dependencies with an unlimited temporal range, working with features provided by the frozen VGG network trained with 16-frame clips.

CONCLUSION AND RECOMMENDATIONS

We studied video recognition with the prism of unified time and space dimensions, forming 3D volumes of video data. This approach has been shown interesting but challenging. Some issues can be attributed to the systems, while others are probably inherent to the task. Deep learning models could benefit from more training data. This is especially true for 3D CNNs. This makes the use of pretraining necessary for FER. The inflation method allows to easily leverage knowledge from 2D models pretrained on images. However, the resulting 3D model is biased towards spatial feature extraction.

We developed and evaluated an efficient training-clip sampling method, to learn and infer from the relevant time-windows. Focusing on informative frames reduces the effect of occlusion, movement, illumination and other issues, and thus allows to better leverage spatiotemporal features to build efficient representations, addressing the complexity of inter-class and intra-class variations, and identity bias. Validating our method on diverse benchmarks, emotion recognition, pain detection, and action recognition, allowed us to identify characteristics of these tasks. We used temporal stochastic softmax for fine-tuning 3D and inflated models, but it could be much more efficient in the pretraining phase if large enough temporal datasets were available, e.g. on action recognition. We designed this temporal stochastic softmax method within the commonly accepted framework of training-clip sampling. However, the idea of clipping videos in time is not in line with the paradigm of unified spatial and temporal dimensions.

As another way to improve recognition performance, attention mechanisms revealed very uneven. We explored different options and proposed adaptations for the spatiotemporal framework. We identified specificities of each type of attention, for the spatiotemporal dimensions and for the task.

Literature shows that 2D models are easier to operate and are still more efficient than 3D models if training data and computational resources are not sufficient for their proper deployment.

Several issues have been identified for 3D CNNs on FER. Extracting low-level dynamics from face videos requires robust face alignement. For real-world data, with occlusion and head-pose variations, an optimal face-alignment system would still not be able to correct all issues. More complex methods are being developed, as face synthesis for pose correction.

Inflation of pretrained models is very efficient to leverage larger 2D FER datasets for spatiotemporal recognition, even though the recognition is biased towards spatial features. Also, alternatives to clip-sampling could be developed in order to allow 3D models to train with longer videos, and learn longer-range dependencies. Given these current issues, conceiving lighter models could also be relevant. Interesting mechanisms as STN could even be used to develop data-driven approaches based on the analysis of discriminant facial parts, that could be easier to train than systems working with a global approach.

APPENDIX I

ARCHITECTURE DETAILS OF 3D-RESNET MODELS

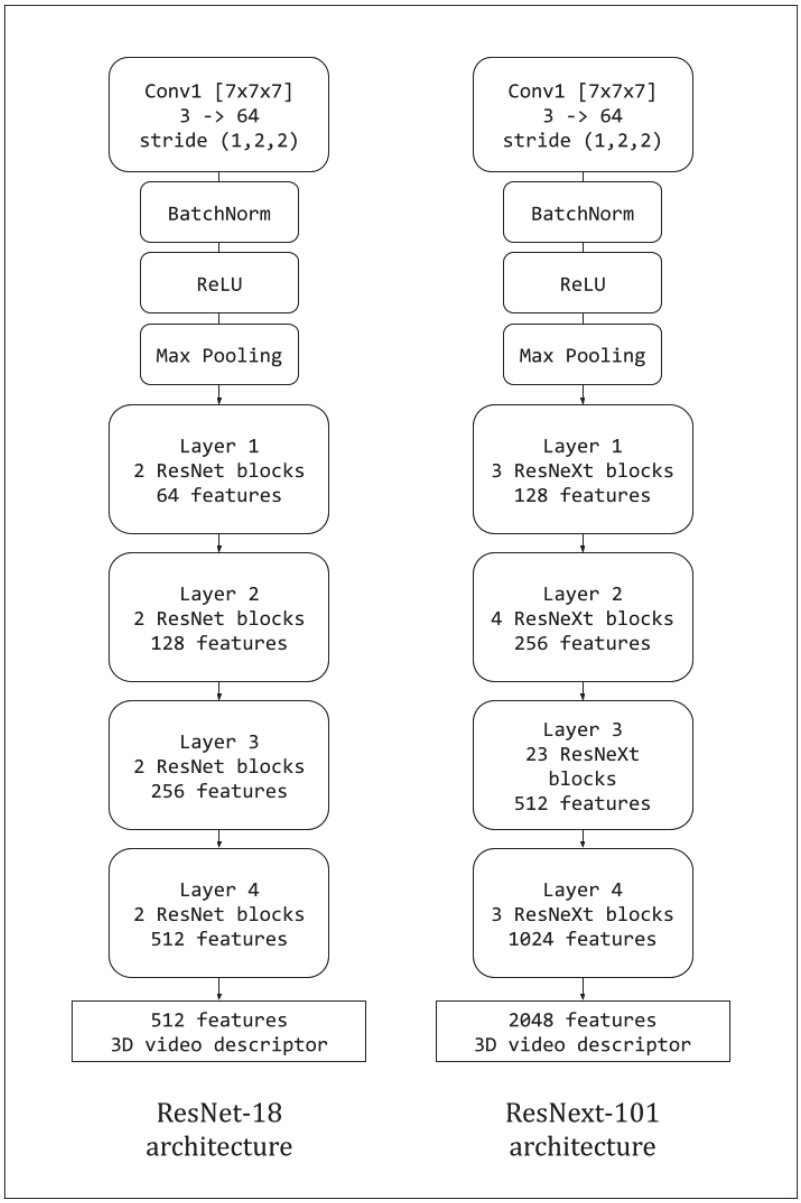

Figure-A I-1 Architecture of the ResNet-18 and ResNext-101 networks (Hara *et al.*, 2018), with blocks containing several convolutional layers, with residual connections, as described in Section 2.1.1

BIBLIOGRAPHY

Abbasnejad, I., Sridharan, S., Tien, D. N., Denman, S., Fookes, C. & Lucey, S. (2017). Using Synthetic Data to Improve Facial Expression Analysis with 3D Convolutional Networks. *ICCV Workshops*, pp. 1609–1618.

Aksan, E., Cao, P., Kaufmann, M. & Hilliges, O. (2020). Attention, please: A Spatio-temporal Transformer for 3D Human Motion Prediction. *CoRR*, abs/2004.08692.

Alexandre, G. R., Soares, J. M. & Thé, G. A. P. (2020). Systematic review of 3D facial expression recognition methods. *Pattern Recognit.*, 100, 107108.

Aminbeidokhti, M., Pedersoli, M., Cardinal, P. & Granger, E. (2019). Emotion Recognition with Spatial Attention and Temporal Softmax Pooling. *ICIAR (1)*, 11662(Lecture Notes in Computer Science), 323–331.

Ayral, T., Pedersoli, M., Bacon, S. & Granger, E. (2021, January). Temporal Stochastic Softmax for 3D CNNs: An Application in Facial Expression Recognition. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 3029-3038.

Ba, J. (2020). *Learning to Attend with Neural Networks*. (Ph.D. thesis, University of Toronto).

Ba, J., Mnih, V. & Kavukcuoglu, K. (2015). Multiple Object Recognition with Visual Attention. *ICLR (Poster)*.

Baccouche, M., Mamalet, F., Wolf, C., Garcia, C. & Baskurt, A. (2011). Sequential Deep Learning for Human Action Recognition. *HBU*, 7065(Lecture Notes in Computer Science), 29–39.

Bahdanau, D., Cho, K. & Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. *ICLR*.

Bargal, S. A., Barsoum, E., Canton-Ferrer, C. & Zhang, C. (2016). Emotion recognition in the wild from videos using images. *ICMI*, pp. 433–436.

Bassili, J. N. (1978). Facial motion in the perception of faces and of emotional expression. *Journal of experimental psychology: human perception and performance*, 4(3), 373.

Bengio, E., Bacon, P., Pineau, J. & Precup, D. (2015). Conditional Computation in Neural Networks for faster models. *CoRR*, abs/1511.06297.

Benitez-Quiroz, C. F., Srinivasan, R. & Martínez, A. M. (2016). EmotioNet: An Accurate, Real-Time Algorithm for the Automatic Annotation of a Million Facial Expressions in the

Wild. *CVPR*, pp. 5562–5570.

Bhuiyan, A., Liu, Y., Siva, P., Javan, M., Ayed, I. B. & Granger, E. (2020). Pose Guided Gated Fusion for Person Re-identification. *WACV*, pp. 2664–2673.

Boureau, Y., Ponce, J. & LeCun, Y. (2010). A Theoretical Analysis of Feature Pooling in Visual Recognition. *ICML 2010*, pp. 111–118.

Britz, D., Goldie, A., Luong, M. & Le, Q. V. (2017). Massive Exploration of Neural Machine Translation Architectures. *CoRR*, abs/1703.03906.

Bulagang, A. F., Weng, N. G., Mountstephens, J. & Teo, J. (2020). A review of recent approaches for emotion classification using electrocardiography and electrodermography signals. *Informatics in Medicine Unlocked*, 20, 100363.

Caballero, J., Ledig, C., Aitken, A. P., Acosta, A., Totz, J., Wang, Z. & Shi, W. (2017). Real-Time Video Super-Resolution with Spatio-Temporal Networks and Motion Compensation. *CVPR*, pp. 2848–2857.

Carreira, J. & Zisserman, A. (2017). Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. *CVPR 2017*, pp. 4724–4733.

Chen, L.-F., Zhou, M., Su, W., Wu, M., She, J. & Hirota, K. (2018a). Softmax regression based deep sparse autoencoder network for facial emotion recognition in human-robot interaction. *Inf. Sci.*, 428, 49–61.

Chen, Z., Ansari, R. & Wilkie, D. J. (2018b). Automated Pain Detection from Facial Expressions using FACS: A Review. *CoRR*, abs/1811.07988.

Cheng, J., Dong, L. & Lapata, M. (2016). Long Short-Term Memory-Networks for Machine Reading. *EMNLP*, pp. 551–561.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *EMNLP*, pp. 1724–1734.

Choi, H., Cho, K. & Bengio, Y. (2018). Fine-grained attention mechanism for neural machine translation. *Neurocomputing*, 284, 171–176.

Ciresan, D. C., Meier, U. & Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. *CVPR*, pp. 3642–3649.

Cohn, J. F. & Ekman, P. (2002). Measuring facial action. In J. A. Harrigan,R. Rosenthal, . K. R. S. (Ed.), *Series in Affective Science. The new handbook of methods in nonverbal behavior research* (pp. 9–64). New York: Oxford University Press.

Corbetta, M. & Shulman, G. (2002). Control of Goal-Directed and Stimulus-Driven Attention in the Brain. *Nature reviews. Neuroscience*, 3, 201-15. doi: 10.1038/nrn755.

Cornia, M., Baraldi, L., Serra, G. & Cucchiara, R. (2016). A deep multi-level network for saliency prediction. *ICPR*, pp. 3488–3493.

Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H. & Wei, Y. (2017). Deformable Convolutional Networks. *ICCV*, pp. 764–773.

Dai, J., Zhang, P., Wang, D., Lu, H. & Wang, H. (2019). Video Person Re-Identification by Temporal Residual Learning. *IEEE Trans. Image Process.*, 28(3), 1366–1377.

Dai, R., Das, S., Minciullo, L., Garattoni, L., Francesca, G. & Bremond, F. (2021, January). PDAN: Pyramid Dilated Attention Network for Action Detection. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 2970-2979.

Darwin, C. (1998). *The expression of the emotions in man and animals*. Oxford University Press.

Das, S., Chaudhary, A., Brémond, F. & Thonnat, M. (2019). Where to Focus on for Human Action Recognition? *WACV*, pp. 71–80.

De Melo, W. C., Granger, E. & Abdenour, H. (2020a). A Deep Multiscale Spatiotemporal Network for Assessing Depression from Facial Dynamics. *IEEE Transactions on Affective Computing*.

De Melo, W. C., Granger, E. & López, M. B. (2020b). Encoding Temporal Information For Automatic Depression Recognition From Facial Analysis. *ICASSP*, pp. 1080–1084.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. *CVPR*, pp. 248–255.

Dhall, A. (2019). EmotiW 2019: Automatic Emotion, Engagement and Cohesion Prediction Tasks. *ICMI*, pp. 546–550.

Dhall, A., Goecke, R., Lucey, S. & Gedeon, T. (2012). Collecting Large, Richly Annotated Facial-Expression Databases from Movies. *IEEE Multim.*, 19(3), 34–41.

Dhall, A., Goecke, R., Joshi, J., Wagner, M. & Gedeon, T. (2013). Emotion recognition in the wild challenge 2013. *ICMI*, pp. 509–516.

Dhall, A., Göcke, R., Joshi, J., Hoey, J. & Gedeon, T. (2016). EmotiW 2016: video and group-level emotion recognition challenges. *ICMI*, pp. 427–432.

Dhall, A., Goecke, R., Ghosh, S., Joshi, J., Hoey, J. & Gedeon, T. (2017). From individual to group-level emotion recognition: EmotiW 5.0. *ICMI*, pp. 524–528.

Dhall, A., Kaur, A., Goecke, R. & Gedeon, T. (2018). EmotiW 2018: Audio-Video, Student Engagement and Group-Level Affect Prediction. *ICMI*, pp. 653–656.

Diba, A., Fayyaz, M., Sharma, V., H., K. A., Arzani, M. M., Yousefzadeh, R. & Van Gool, L. (2018). Temporal 3D ConvNets Using Temporal Transition Layer. *CVPR Workshops*, pp. 1117–1121.

Diba, A., Fayyaz, M., Sharma, V., Karami, A. H., Arzani, M. M., Yousefzadeh, R. & Van Gool, L. (2017). Temporal 3D ConvNets: New Architecture and Transfer Learning for Video Classification. *CoRR*, abs/1711.08200.

Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Darrell, T. & Saenko, K. (2015). Long-term recurrent convolutional networks for visual recognition and description. *CVPR*, pp. 2625–2634.

Du, S., Tao, Y. & Martínez, A. M. (2014). Compound facial expressions of emotion. *Proc. Natl. Acad. Sci. USA*, 111(15), E1454–E1462.

Dyer, J. & Kolic, B. (2020). Public risk perception and emotion on Twitter during the Covid-19 pandemic. *Appl. Netw. Sci.*, 5(1), 99.

Ekman, P. & Friesen, W. (2003). *Unmasking the Face: A Guide to Recognizing Emotions from Facial Clues*. Malor Books. Consulted https://books.google.ca/books?id=TukNoJDgMTUC.

Ekman, P., Friesen, W. V. & Hager, J. C. (2002). Facial Action Coding System. Manual and Investigator's Guide. *Research Nexus*.

Ekman, P. & Friesen, W. V. (1969). The repertoire of nonverbal behavior: Categories, origins, usage, and coding. *Nonverbal communication, interaction, and gesture*, 57–106.

Ekman, P. & Friesen, W. V. (1971). Constants across cultures in the face and emotion. *Journal of personality and social psychology*, 17(2), 124.

Fan, Y., Lu, X., Li, D. & Liu, Y. (2016). Video-based emotion recognition using CNN-RNN and C3D hybrid networks. *ICMI*, pp. 445–450.

Fan, Y., Lam, J. C. K. & Li, V. O. K. (2018). Video-based Emotion Recognition Using Deeply-Supervised Neural Networks. *ICMI*, pp. 584–588.

Freiwald, W. A., Tsao, D. Y. & Livingstone, M. S. (2009). A face feature space in the macaque temporal lobe. *Nature neuroscience*, 12(9), 1187–1196.

Gabbiani, F., Krapp, H. G., Koch, C. & Laurent, G. (2002). Multiplicative computation in a visual neuron sensitive to looming. *Nature*, 420(6913), 320–324.

Gao, B. & Pavel, L. (2017). On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning. *CoRR*, abs/1704.00805.

Girdhar, R., Carreira, J., Doersch, C. & Zisserman, A. (2019). Video Action Transformer Network. *CVPR*, pp. 244–253.

Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, A. C., Mirza, M., Hamner, B., Cukierski, W., Tang, Y., Thaler, D., Lee, D., Zhou, Y., Ramaiah, C., Feng, F., Li, R., Wang, X., Athanasakis, D., Shawe-Taylor, J., Milakov, M., Park, J., Ionescu, R. T., Popescu, M., Grozea, C., Bergstra, J., Xie, J., Romaszko, L., Xu, B., Zhang, C. & Bengio, Y. (2013). Challenges in Representation Learning: A Report on Three Machine Learning Contests. *ICONIP (3)*, 8228(Lecture Notes in Computer Science), 117–124.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C. & Bengio, Y. (2014). Generative Adversarial Nets. *NIPS*, pp. 2672–2680.

Graves, A., Wayne, G. & Danihelka, I. (2014). Neural Turing Machines. *CoRR*, abs/1410.5401.

Gregor, K., Danihelka, I., Graves, A., Rezende, D. J. & Wierstra, D. (2015). DRAW: A Recurrent Neural Network For Image Generation. *ICML*, 37(JMLR Workshop and Conference Proceedings), 1462–1471.

Gunes, H. & Schuller, B. W. (2013). Categorical and dimensional affect analysis in continuous input: Current trends and future directions. *Image Vis. Comput.*, 31(2), 120–136.

Hara, K., Kataoka, H. & Satoh, Y. (2018). Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet? *CVPR 2018*, pp. 6546–6555.

Hayhoe, M. & Ballard, D. (2005). Eye Movements in Natural Behavior. *Trends in cognitive sciences*, 9, 188-94. doi: 10.1016/j.tics.2005.02.009.

He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep Residual Learning for Image Recognition. *CVPR*, pp. 770–778.

He, S., Wang, S., Lan, W., Fu, H. & Ji, Q. (2013). Facial Expression Recognition Using Deep Boltzmann Machine from Thermal Infrared Images. *ACII*, pp. 239–244.

Hinton, G. E. & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, 313(5786), 504–507.

Hinton, G. E., Sejnowski, T. J. et al. (1986). Learning and relearning in Boltzmann machines. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1(282-317), 2.

Hinton, G. E., Osindero, S. & Teh, Y. W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Comput.*, 18(7), 1527–1554.

Hinton, G. E., Vinyals, O. & Dean, J. (2015). Distilling the Knowledge in a Neural Network. *CoRR*, abs/1503.02531.

Hochreiter, S. & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Comput.*, 9(8), 1735–1780.

Hou, R., Chen, C. & Shah, M. (2017). An End-to-end 3D Convolutional Neural Network for Action Detection and Segmentation in Videos. *CoRR*, abs/1712.01111.

Hu, J., Shen, L. & Sun, G. (2018). Squeeze-and-Excitation Networks. *CVPR*, pp. 7132–7141.

Hu, J., Shen, L., Albanie, S., Sun, G. & Wu, E. (2020). Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(8), 2011–2023.

Huang, C. A., Vaswani, A., Uszkoreit, J., Simon, I., Hawthorne, C., Shazeer, N., Dai, A. M., Hoffman, M. D., Dinculescu, M. & Eck, D. (2019). Music Transformer: Generating Music with Long-Term Structure. *ICLR (Poster)*.

Huang, D.-A., Ramanathan, V., Mahajan, D., Torresani, L., Paluri, M., Fei-Fei, L. & Niebles, J. C. (2018). What Makes a Video a Video: Analyzing Temporal Information in Video Understanding Models and Datasets. *CVPR*, pp. 7366–7375.

Hubel, D. H. & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1), 106–154.

Ioffe, S. & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *ICML*, 37(JMLR Workshop and Conference Proceedings),

448–456.

Itti, L., Koch, C. & Niebur, E. (1998). A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(11), 1254–1259.

Jack, R. E., Garrod, O. G., Yu, H., Caldara, R. & Schyns, P. G. (2012). Facial expressions of emotion are not culturally universal. *Proceedings of the National Academy of Sciences*, 109(19), 7241–7244.

Jaderberg, M., Simonyan, K., Zisserman, A. & Kavukcuoglu, K. (2015). Spatial Transformer Networks. *NIPS*, pp. 2017–2025.

Jeni, L. A., Girard, J. M., Cohn, J. F. & la Torre, F. D. (2013). Continuous AU intensity estimation using localized, sparse facial feature space. *FG*, pp. 1–7.

Ji, S., Xu, W., Yang, M. & Yu, K. (2010). 3D Convolutional Neural Networks for Human Action Recognition. *ICML*, pp. 495–502.

Johansson, G. (1973). Visual perception of biological motion and a model for its analysis. *Perception & psychophysics*, 14(2), 201–211.

Jung, H., Lee, S., Yim, J., Park, S. & Kim, J. (2015). Joint Fine-Tuning in Deep Neural Networks for Facial Expression Recognition. *ICCV*, pp. 2983–2991.

Kächele, M., Schels, M., Meudt, S., Palm, G. & Schwenker, F. (2016). Revisiting the EmotiW challenge: how wild is it really? *J. Multimodal User Interfaces*, 10(2), 151–162.

Kahou, S. E., Michalski, V., Konda, K. R., Memisevic, R. & Pal, C. J. (2015). Recurrent Neural Networks for Emotion Recognition in Video. *ICMI*, pp. 467–474.

Kahou, S. E., Michalski, V., Memisevic, R., Pal, C. J. & Vincent, P. (2017). RATM: Recurrent Attentive Tracking Model. *CVPR Workshops*, pp. 1613–1622.

Kaiser, L., Gomez, A. N., Shazeer, N., Vaswani, A., Parmar, N., Jones, L. & Uszkoreit, J. (2017). One Model To Learn Them All. *CoRR*, abs/1706.05137.

Kamarol, S. K. A., Jaward, M. H., Kälviäinen, H., Parkkinen, J. & Parthiban, R. (2017). Joint facial expression recognition and intensity estimation based on weighted votes of image sequences. *Pattern Recognit. Lett.*, 92, 25–32.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R. & Fei-Fei, L. (2014). Large-Scale Video Classification with Convolutional Neural Networks. *CVPR*, pp. 1725–1732.

Katharopoulos, A. & Fleuret, F. (2018). Not All Samples Are Created Equal: Deep Learning with Importance Sampling. *ICML*, 80(Proceedings of Machine Learning Research), 2530–2539.

Kaur, A., Mustafa, A., Mehta, L. & Dhall, A. (2018). Prediction and Localization of Student Engagement in the Wild. *DICTA*, pp. 1–8.

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M. & Zisserman, A. (2017). The Kinetics Human Action Video Dataset. *CoRR*, abs/1705.06950.

Kaya, H., Gürpinar, F. & Salah, A. A. (2017). Video-based emotion recognition in the wild using deep transfer learning and score fusion. *Image Vis. Comput.*, 65, 66–75.

Kim, T. H., Sajjadi, M. S. M., Hirsch, M. & Schölkopf, B. (2018). Spatio-Temporal Transformer Network for Video Restoration. *ECCV (3)*, 11207(Lecture Notes in Computer Science), 111–127.

Knyazev, B., Shvetsov, R., Efremova, N. & Kuharenko, A. (2017). Convolutional neural networks pretrained on large face recognition datasets for emotion classification from video. *CoRR*, abs/1711.04598.

Knyazev, B., Shvetsov, R., Efremova, N. & Kuharenko, A. (2018). Leveraging Large Face Recognition Data for Emotion Classification. *FG*, pp. 692–696.

Korbar, B., Tran, D. & Torresani, L. (2019). SCSampler: Sampling Salient Clips From Video for Efficient Action Recognition. *ICCV*, pp. 6231–6241.

Kortylewski, A., Schneider, A., Gerig, T., Egger, B., Morel-Forster, A. & Vetter, T. (2018). Training Deep Face Recognition Systems with Synthetic Data. *CoRR*, abs/1802.05891.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *NIPS*, pp. 1106–1114.

Kuehne, H., Jhuang, H., Garrote, E., Poggio, T. A. & Serre, T. (2011). HMDB: A large video database for human motion recognition. *ICCV*, pp. 2556–2563.

Kumar, V., Rao, S. & Yu, L. (2020). Noisy Student Training Using Body Language Dataset Improves Facial Expression Recognition. *ECCV Workshops (1)*, 12535(Lecture Notes in Computer Science), 756–773.

Kumawat, S., Verma, M. & Raman, S. (2019). LBVCNN: Local Binary Volume Convolutional Neural Network for Facial Expression Recognition From Image Sequences. *CVPR Workshops*, pp. 207–216.

Lai, Y. & Lai, S. (2018). Emotion-Preserving Representation Learning via Generative Adversarial Network for Multi-View Facial Expression Recognition. *FG*, pp. 263–270.

Laptev, I., Marszalek, M., Schmid, C. & Rozenfeld, B. (2008). Learning realistic human actions from movies. *CVPR*.

Larochelle, H. & Hinton, G. E. (2010). Learning to combine foveal glimpses with a third-order Boltzmann machine. *NIPS*, pp. 1243–1251.

Lecun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.

Lee, J., Kim, S., Kim, S. & Sohn, K. (2018). Spatiotemporal Attention Based Deep Neural Networks for Emotion Recognition. *ICASSP*, pp. 1513–1517.

Li, S. & Deng, W. (2018). Deep Facial Expression Recognition: A Survey. *CoRR*, abs/1804.08348.

Li, S., Bak, S., Carr, P. & Wang, X. (2018a). Diversity Regularized Spatiotemporal Attention for Video-Based Person Re-Identification. *CVPR*, pp. 369–378.

Li, S., Zheng, W., Zong, Y., Lu, C., Tang, C., Jiang, X., Liu, J. & Xia, W. (2019). Bi-modality Fusion for Emotion Recognition in the Wild. *ICMI*, pp. 589–594.

Li, W., Huang, D., Li, H. & Wang, Y. (2018b). Automatic 4D Facial Expression Recognition Using Dynamic Geometrical Image Network. *FG*, pp. 24–30.

Lin, M., Chen, Q. & Yan, S. (2013). Network in network. *arXiv preprint arXiv:1312.4400*.

Lin, Z., Feng, M., dos Santos, C. N., Yu, M., Xiang, B., Zhou, B. & Bengio, Y. (2017). A Structured Self-Attentive Sentence Embedding. *ICLR (Poster)*.

Liu, C., Tang, T., Lv, K. & Wang, M. (2018a). Multi-Feature Based Emotion Recognition for Video Clips. *ICMI*, pp. 630–634.

Liu, K., Liu, W., Gan, C., Tan, M. & Ma, H. (2018b). T-C3D: Temporal Convolutional 3D Network for Real-Time Action Recognition. *AAAI*, pp. 7138–7145.

Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L. & Shazeer, N. (2018c). Generating Wikipedia by Summarizing Long Sequences. *ICLR (Poster)*.

Liu, X., Kan, M., Wu, W., Shan, S. & Chen, X. (2017). VIPLFaceNet: an open source deep face recognition SDK. *Frontiers Comput. Sci.*, 11(2), 208–218.

Lu, C., Zheng, W., Li, C., Tang, C., Liu, S., Yan, S. & Zong, Y. (2018). Multiple Spatio-temporal Feature Learning for Video-based Emotion Recognition in the Wild. *ICMI*, pp. 646–652.

Lucey, P., Cohn, J. F., Prkachin, K. M., Solomon, P. E. & Matthews, I. A. (2011). Painful data: The UNBC-McMaster shoulder pain expression archive database. *FG*, pp. 57–64.

Luo, Z., Chen, J., Takiguchi, T. & Ariki, Y. (2017). Facial Expression Recognition with deep age. *ICME Workshops*, pp. 657–662.

Luong, T., Pham, H. & Manning, C. D. (2015). Effective Approaches to Attention-based Neural Machine Translation. *EMNLP*, pp. 1412–1421.

Lyons, M. J., Akamatsu, S., Kamachi, M. & Gyoba, J. (1998). Coding Facial Expressions with Gabor Wavelets. *FG*, pp. 200–205.

Mathe, S. & Sminchisescu, C. (2013). (2013). Action from still image dataset and inverse optimal control to learn task specific visual scanpaths. *In Advances in neural information processing systems*, 1923-1931.

Mavani, V., Raman, S. & Miyapuram, K. P. (2017). Facial Expression Recognition Using Visual Saliency and Deep Learning. *ICCV Workshops*, pp. 2783–2788.

McFee, B., Salamon, J. & Bello, J. P. (2018). Adaptive Pooling Operators for Weakly Labeled Sound Event Detection. *IEEE ACM Trans. Audio Speech Lang. Process.*, 26(11), 2180–2193.

Memisevic, R. (2011). Learning to relate images: Mapping units, complex cells and simultaneous eigenspaces. *CoRR*, abs/1110.0107.

Memisevic, R. (2013). Learning to Relate Images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8), 1829–1846.

Meng, D., Peng, X., Wang, K. & Qiao, Y. (2019). Frame Attention Networks for Facial Expression Recognition in Videos. *ICIP*, pp. 3866–3870.

Miech, A., Laptev, I. & Sivic, J. (2017). Learnable pooling with Context Gating for video classification. *CoRR*, abs/1706.06905.

Misra, I., Zitnick, C. L. & Hebert, M. (2016). Shuffle and Learn: Unsupervised Learning Using Temporal Order Verification. *ECCV (1)*, 9905(Lecture Notes in Computer Science), 527–544.

Mnih, V., Heess, N., Graves, A. & Kavukcuoglu, K. (2014). Recurrent Models of Visual Attention. *NIPS*, pp. 2204–2212.

Mollahosseini, A., Hassani, B. & Mahoor, M. H. (2019). AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild. *IEEE Trans. Affect. Comput.*, 10(1), 18–31.

Ng, J. Y., Hausknecht, M. J., Vijayanarasimhan, S., Vinyals, O., Monga, R. & Toderici, G. (2015). Beyond short snippets: Deep networks for video classification. *CVPR*, pp. 4694–4702.

Niebles, J. C., Wang, H. & Fei-Fei, L. (2008). Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words. *Int. J. Comput. Vis.*, 79(3), 299–318.

Othman, E., Werner, P., Saxen, F., Al-Hamadi, A. & Walter, S. (2019). Cross-Database Evaluation of Pain Recognition from Facial Video. *ISPA*, pp. 181–186.

Ouyang, X., Kawaai, S., Goh, E. G. H., Shen, S., Ding, W., Ming, H. & Huang, D. (2017). Audio-visual emotion recognition using deep transfer learning and multiple temporal models. *ICMI*, pp. 577–582.

Pantic, M. & Rothkrantz, L. J. M. (2000a). Expert system for automatic analysis of facial expressions. *Image Vis. Comput.*, 18(11), 881–905.

Pantic, M. & Rothkrantz, L. J. M. (2000b). Automatic Analysis of Facial Expressions: The State of the Art. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12), 1424–1445.

Parkhi, O. M., Vedaldi, A. & Zisserman, A. (2015). Deep Face Recognition. *BMVC*, pp. 41.1–41.12.

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A. & Tran, D. (2018). Image Transformer. *ICML*, 80(Proceedings of Machine Learning Research), 4052–4061.

Parmar, N., Ramachandran, P., Vaswani, A., Bello, I., Levskaya, A. & Shlens, J. (2019). Stand-Alone Self-Attention in Vision Models. *NeurIPS*, pp. 68–80.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. & Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. *NeurIPS*, pp. 8024–8035.

Petersen, S. & Posner, M. (2012). The Attention System of the Human Brain: 20 Years After. *Annual review of neuroscience*, 35, 73-89. doi: 10.1146/annurev-neuro-062111-150525.

Posner, M. I. & Dehaene, S. (1994). Attentional networks. *Trends in Neurosciences*, 17(2), 75 - 79. doi: https://doi.org/10.1016/0166-2236(94)90078-7.

Praveen, G., Granger, E. & Cardinal, P. (2019). Deep Weakly-Supervised Domain Adaptation for Pain Localization in Videos. *FG*.

Rahman, T., Rochan, M. & Wang, Y. (2019). Video-Based Person Re-Identification using Refined Attention Networks. *AVSS*, pp. 1–8.

Ranzato, M., Mnih, V., Susskind, J. M. & Hinton, G. E. (2013). Modeling Natural Images Using Gated MRFs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(9), 2206–2222.

Ringeval, F. (2011). *Ancrages et modèles dynamiques de la prosodie: application à la reconnaissance des émotions actées et spontanées*. (Ph.D. thesis, Université Pierre et Marie Curie-Paris VI).

Russell, J. A. (1980). A circumplex model of affect. *Journal of personality and social psychology*, 39(6), 1161.

Sariyanidi, E., Gunes, H. & Cavallaro, A. (2015). Automatic Analysis of Facial Affect: A Survey of Registration, Representation, and Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(6), 1113–1133.

Schindler, K. & Van Gool, L. (2008). Action snippets: How many frames does human action recognition require? *CVPR*.

Sermanet, P., Frome, A. & Real, E. (2015). Attention for Fine-Grained Categorization. *ICLR (Workshop)*.

Sevilla-Lara, L., Liao, Y., Güney, F., Jampani, V., Geiger, A. & Black, M. J. (2018). On the Integration of Optical Flow and Action Recognition. *GCPR*, 11269(Lecture Notes in Computer Science), 281–297.

Sevilla-Lara, L., Zha, S., Yan, Z., Goswami, V., Feiszli, M. & Torresani, L. (2019). Only Time Can Tell: Discovering Temporal Data for Temporal Modeling. *CoRR*, abs/1907.08340.

Shan, C., Gong, S. & McOwan, P. W. (2009). Facial expression recognition based on Local Binary Patterns: A comprehensive study. *Image Vis. Comput.*, 27(6), 803–816.

Shaw, P., Uszkoreit, J. & Vaswani, A. (2018). Self-Attention with Relative Position Representations. *NAACL-HLT (2)*, pp. 464–468.

Sikka, K. & Sharma, G. (2018). Discriminatively Trained Latent Ordinal Model for Video Classification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(8), 1829–1844.

Simonyan, K. & Zisserman, A. (2014). Two-Stream Convolutional Networks for Action Recognition in Videos. *NIPS*, pp. 568–576.

Simonyan, K. & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ICLR*.

Sivic, J. & Zisserman, A. (2003). Video Google: A Text Retrieval Approach to Object Matching in Videos. *ICCV*, pp. 1470–1477.

Springenberg, J. T., Dosovitskiy, A., Brox, T. & Riedmiller, M. A. (2015). Striving for Simplicity: The All Convolutional Net. *ICLR (Workshop)*.

Sukhbaatar, S., Szlam, A., Weston, J. & Fergus, R. (2015). End-To-End Memory Networks. *NIPS*, pp. 2440–2448.

Sun, L., Jia, K., Yeung, D. & Shi, B. E. (2015). Human Action Recognition Using Factorized Spatio-Temporal Convolutional Networks. *ICCV*, pp. 4597–4605.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V. & Rabinovich, A. (2015). Going deeper with convolutions. *CVPR*, pp. 1–9.

Tang, Y. (2013). Deep learning using linear support vector machines. *arXiv preprint arXiv:1306.0239*.

Tavakolian, M. & Hadid, A. (2019). A Spatiotemporal Convolutional Neural Network for Automatic Pain Intensity Estimation from Facial Dynamics. *Int. J. Comput. Vis.*, 127(10), 1413–1425.

Taylor, G. W., Fergus, R., LeCun, Y. & Bregler, C. (2010). Convolutional Learning of Spatio-temporal Features. *ECCV (6)*, 6316(Lecture Notes in Computer Science), 140–153.

Thiam, P., Kestler, H. A. & Schwenker, F. (2020). Two-Stream Attention Network for Pain Recognition from Video Sequences. *Sensors*, 20(3), 839.

Torralba, A., Oliva, A., Castelhano, M. & Henderson, J. (2006). Contextual Guidance of Eye Movements and Attention in Real-World Scenes: The Role of Global Features in Object Search. *Psychological review*, 113, 766-86. doi: 10.1037/0033-295X.113.4.766.

Tran, D., Jamie Ray, J., Zheng Shou, Z., Chang, S. & Paluri, M. (2017). ConvNet Architecture Search for Spatiotemporal Feature Learning. *CoRR*, abs/1708.05038.

Tran, D., Bourdev, L. D., Fergus, R., Torresani, L. & Paluri, M. (2015). Learning Spatiotemporal Features with 3D Convolutional Networks. *ICCV*, pp. 4489–4497.

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y. & Paluri, M. (2018). A Closer Look at Spatiotemporal Convolutions for Action Recognition. *CVPR*, pp. 6450–6459.

Valstar, M. F., Schuller, B. W., Smith, K., Eyben, F., Jiang, B., Bilakhia, S., Schnieder, S., Cowie, R. & Pantic, M. (2013). AVEC 2013: the continuous audio/visual emotion and depression recognition challenge. *AVEC@ACM Multimedia*, pp. 3–10.

Valstar, M. F. & Pantic, M. (2012). Fully Automatic Recognition of the Temporal Phases of Facial Actions. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, 42(1), 28–43.

Valstar, M. F., Jiang, B., Mehu, M., Pantic, M. & Scherer, K. R. (2011). The first facial expression recognition and analysis challenge. *FG*, pp. 921–926.

Varol, G., Laptev, I. & Schmid, C. (2018). Long-Term Temporal Convolutions for Action Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(6), 1510–1517.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. (2017). Attention is All you Need. *NIPS*, pp. 5998–6008.

Veit, A. & Belongie, S. J. (2020). Convolutional Networks with Adaptive Inference Graphs. *Int. J. Comput. Vis.*, 128(3), 730–741.

Vielzeuf, V. (2019). *Apprentissage neuronal profond pour l'analyse de contenus multimodaux et temporels. (Deep learning for multimodal and temporal contents analysis)*. (Ph.D. thesis, Normandy University, Caen, France).

Vielzeuf, V., Pateux, S. & Jurie, F. (2017). Temporal multimodal fusion for video emotion classification in the wild. *ICMI*, pp. 569–576.

Vielzeuf, V., Kervadec, C., Pateux, S., Lechervy, A. & Jurie, F. (2018). An Occam's Razor View on Learning Audiovisual Emotion Recognition with Small Training Sets. *ICMI*, pp. 589–593.

Viola, P. A. & Jones, M. J. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. *CVPR (1)*, pp. 511–518.

Wan, J., Escalera, S., Anbarjafari, G., Escalante, H. J., Baró, X., Guyon, I., Madadi, M., Allik, J., Gorbova, J., Lin, C. & Xie, Y. (2017). Results and Analysis of ChaLearn LAP Multi-modal Isolated and Continuous Gesture Recognition, and Real Versus Fake Expressed Emotions Challenges. *ICCV Workshops*, pp. 3189–3197.

Wang, H., Ullah, M. M., Kläser, A., Laptev, I. & Schmid, C. (2009). Evaluation of Local Spatio-temporal Features for Action Recognition. *BMVC*, pp. 1–11.

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B. & Wu, Y. (2014). Learning Fine-Grained Image Similarity with Deep Ranking. *CVPR*, pp. 1386–1393.

Wang, L., Li, W., Li, W. & Van Gool, L. (2018a). Appearance-and-Relation Networks for Video Classification. *CVPR*, pp. 1430–1439.

Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X. & Van Gool, L. (2016). Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. *ECCV (8)*, 9912(Lecture Notes in Computer Science), 20–36.

Wang, X., Girshick, R. B., Gupta, A. & He, K. (2018b). Non-Local Neural Networks. *CVPR*, pp. 7794–7803.

Wang, Y., Yu, H., Stevens, B. & Liu, H. (2015). Dynamic facial expression recognition using local patch and LBP-TOP. *HSI*, pp. 362–367.

Werner, P., Al-Hamadi, A., Niese, R., Walter, S., Gruss, S. & Traue, H. C. (2013). Towards Pain Monitoring: Facial Expression, Head Pose, a new Database, an Automatic System and Remaining. *BMVC*.

Werner, P., Al-Hamadi, A., Limbrecht-Ecklundt, K., Walter, S., Gruss, S. & Traue, H. C. (2017a). Automatic Pain Assessment with Facial Activity Descriptors. *IEEE Trans. Affect. Comput.*, 8(3), 286–299.

Werner, P., Al-Hamadi, A. & Walter, S. (2017b). Analysis of facial expressiveness during experimentally induced heat pain. *ACII Workshops*, pp. 176–180.

Werner, P., Al-Hamadi, A., Limbrecht-Ecklundt, K., Walter, S. & Traue, H. C. (2018). Head movements and postures as pain behavior. *PLOS ONE*, 13(2), 1-17. doi: 10.1371/journal.pone.0192767.

Williams, R. J. & Peng, J. (1990). An Efficient Gradient-Based Algorithm for On-Line Training of Recurrent Network Trajectories. *Neural Computation*, 2(4), 490–501.

Woo, S., Park, J., Lee, J. & Kweon, I. S. (2018). CBAM: Convolutional Block Attention Module. *ECCV (7)*, 11211(Lecture Notes in Computer Science), 3–19.

Wu, C., Wang, S. & Ji, Q. (2015). Multi-instance Hidden Markov Model for facial expression recognition. *FG*, pp. 1–6.

Wu, S., Kan, M., He, Z., Shan, S. & Chen, X. (2017). Funnel-structured cascade for multi-view face detection with alignment-awareness. *Neurocomputing*, 221, 138–145.

Xie, S., Girshick, R. B., Dollár, P., Tu, Z. & He, K. (2017). Aggregated Residual Transformations for Deep Neural Networks. *CVPR*, pp. 5987–5995.

Xie, S., Sun, C., Huang, J., Tu, Z. & Murphy, K. (2018). Rethinking Spatiotemporal Feature Learning: Speed-Accuracy Trade-offs in Video Classification. *ECCV (15)*, 11219(Lecture Notes in Computer Science), 318–335.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A. C., Salakhutdinov, R., Zemel, R. S. & Bengio, Y. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *ICML*, 37(JMLR Workshop and Conference Proceedings), 2048–2057.

Xue, L., Li, X. & Zhang, N. L. (2020). Not All Attention Is Needed: Gated Attention Network for Sequence Data. *AAAI*, pp. 6550–6557.

Yang, R., Tong, S., López, M. B., Boutellaa, E., Peng, J., Feng, X. & Hadid, A. (2016). On pain assessment from facial videos using spatio-temporal local descriptors. *IPTA*, pp. 1–6.

Yang, S., Luo, P., Loy, C. C. & Tang, X. (2015). From Facial Parts Responses to Face Detection: A Deep Learning Approach. *ICCV*, pp. 3676–3684.

Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C. J., Larochelle, H. & Courville, A. C. (2015). Describing Videos by Exploiting Temporal Structure. *ICCV*, pp. 4507–4515.

Yu, C. & Tapus, A. (2019). Interactive Robot Learning for Multimodal Emotion Recognition. *ICSR*, 11876(Lecture Notes in Computer Science), 633–642.

Zagoruyko, S. & Komodakis, N. (2017). Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer. *ICLR (Poster)*.

Zambaldi, V. F., Raposo, D., Santoro, A., Bapst, V., Li, Y., Babuschkin, I., Tuyls, K., Reichert, D. P., Lillicrap, T. P., Lockhart, E., Shanahan, M., Langston, V., Pascanu, R., Botvinick, M., Vinyals, O. & Battaglia, P. W. (2019). Deep reinforcement learning with relational inductive biases. *ICLR (Poster)*.

Zamprogno, M., Passon, M., Martinel, N., Serra, G., Lancioni, G., Micheloni, C., Tasso, C. & Foresti, G. L. (2019). Video-Based Convolutional Attention for Person Re-Identification. *ICIAP (1)*, 11751(Lecture Notes in Computer Science), 3–14.

Zaremba, W. & Sutskever, I. (2015). Reinforcement Learning Neural Turing Machines. *CoRR*, abs/1505.00521.

Zeng, J., Shan, S. & Chen, X. (2018). Facial Expression Recognition with Inconsistently Annotated Datasets. *ECCV (13)*, 11217(Lecture Notes in Computer Science), 227–243.

Zhang, F., Zhang, T., Mao, Q. & Xu, C. (2018). Joint Pose and Expression Modeling for Facial Expression Recognition. *CVPR*, pp. 3359–3368.

Zhang, H., Goodfellow, I. J., Metaxas, D. N. & Odena, A. (2019). Self-Attention Generative Adversarial Networks. *ICML*, 97(Proceedings of Machine Learning Research), 7354–7363.

Zhang, J., Shan, S., Kan, M. & Chen, X. (2014). Coarse-to-Fine Auto-Encoder Networks (CFAN) for Real-Time Face Alignment. *ECCV (2)*, 8690(Lecture Notes in Computer Science), 1–16.

Zhang, K., Zhang, Z., Li, Z. & Qiao, Y. (2016a). Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks. *CoRR*, abs/1604.02878.

Zhang, T., Zheng, W., Cui, Z., Zong, Y., Yan, J. & Yan, K. (2016b). A Deep Neural Network-Driven Feature Learning Method for Multi-view Facial Expression Recognition. *IEEE Trans. Multim.*, 18(12), 2528–2536.

Zhao, G. & Pietikäinen, M. (2007). Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6), 915–928.

Zhao, L., Li, X., Zhuang, Y. & Wang, J. (2017). Deeply-Learned Part-Aligned Representations for Person Re-identification. *ICCV*, pp. 3239–3248.

Zhao, X., Liang, X., Liu, L., Li, T., Han, Y., Vasconcelos, N. & Yan, S. (2016). Peak-Piloted Deep Network for Facial Expression Recognition. *ECCV (2)*, 9906(Lecture Notes in Computer Science), 425–442.

Zhi, R., Flierl, M., Ruan, Q. & Kleijn, W. B. (2011). Graph-Preserving Sparse Nonnegative Matrix Factorization With Application to Facial Expression Recognition. *IEEE Trans. Syst. Man Cybern. Part B*, 41(1), 38–52.

Zhong, L., Liu, Q., Yang, P., Huang, J. & Metaxas, D. N. (2015). Learning Multiscale Active Facial Patches for Expression Analysis. *IEEE Trans. Cybern.*, 45(8), 1499–1510.

Zhou, B., Khosla, A., Lapedriza, À., Oliva, A. & Torralba, A. (2016). Learning Deep Features for Discriminative Localization. *CVPR*, pp. 2921–2929.

Zhou, W., Xu, Y., Simpson, I. & Cai, Y. (2007). Multiplicative computation in the vestibulo-ocular reflex (VOR). *Journal of neurophysiology*, 97(4), 2780–2789.

Zhu, L., Tran, D., Sevilla-Lara, L., Yang, Y., Feiszli, M. & Wang, H. (2020). FASTER Recurrent Networks for Efficient Video Classification. *AAAI*, pp. 13098–13105.

Zhu, W., Hu, J., Sun, G., Cao, X. & Qiao, Y. (2016). A Key Volume Mining Deep Framework for Action Recognition. *CVPR*, pp. 1991–1999.

Zhu, X., Cheng, D., Zhang, Z., Lin, S. & Dai, J. (2019). An Empirical Study of Spatial Attention Mechanisms in Deep Networks. *ICCV*, pp. 6687–6696.