Master Thesis in Computer Science
RHEINISCHE FRIEDRICH-WILHELMS-UNIVERSITÄT BONN
Institut für Informatik III
Computer Vision Group
Prof. Dr. J. Gall

A Study of Audio Effectiveness for Action Recognition in Egocentric Videos

20. July 2020

submission by:
Tridivraj Bhattacharyya
Matriculation Number: 3035538

Examiners:
Prof. Dr. Juergen Gall
Prof. Dr. Christian Bauckhage

Supervisor:
Yaser Souri

Rheinische
Friedrich-Wilhelms-
Universität Bonn

Prüfungsausschuss Informatik

universität **bonn** · Institut für Informatik · 53012 Bonn

Vorsitzender des Prüfungsausschusses

Prof. Dr. Thomas Kesselheim

Prüfungsamt:
Judith König
Tel.: 0228/73-4418
Fax : 0228/73-4788
pa@informatik.uni-bonn.de
Postanschrift: 53115 Bonn
Endenicher Allee 19a

www.informatik.uni-bonn.de

Erklärung über das selbständige Verfassen einer Abschlussarbeit/ Declaration of Authorship

Hiermit versichere ich, _____, _____,
Name / name, Vorname / first name

dass ich die Arbeit – bei einer Gruppenarbeit meinen entsprechend gekennzeichneten Anteil der Arbeit – selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

…… Bonn, den ……..
Unterschrift/signature date

Contents

Abstract

In recent times, research into the fusion of audio-visual modalities for Action Recognition (AR) from videos has been picking up pace. We study the effect audio has as an individual modality and also combined with RGB and Optical Flow when applied to Action Recognition from egocentric videos. We introduce new attention mechanisms that can be used to select the relevant temporal regions of an audio clip while looking at a single RGB frame. We create a general framework to create fusion-based Action Recognition models with audio. It also allows one to easily slot the attention mechanism into the Temporal Binding Network (TBN), which is a special case of our model. We create baselines with the TBN, to fuse audio with other modalities and evaluate our attention experiments on top of them. We run tests on the Epic-Kitchens dataset and our results show improvement of top-1 accuracy scores over the baselines.

Acknowledgements

It would be impossible to remember every person who offered their help, directly and indirectly during this work. Nonetheless, I would first like to thank Professor Juergen Gall for his guidance during the thesis. I would also like to thank the author of the Epic-Fusion [44] paper for the clarifications and tips on his work. It would be amiss not to mention the different people from the Computer Vision Group for their various advice at different points of time. A big thanks to my colleagues from the AStA (Department for International Students) along with friends and batch-mates that I made during my coursework for the continuous mental support throughout my studies and thesis work. Finally, it would be criminal not to mention my trusted supervisor, Yaser Souri, for the thesis. I am very grateful for the countless hours of discussions, debates, guidance and support, especially when the work was starting to look overwhelming. This thesis would not have been possible without him. Thank you to everyone and forgive me if I forgot to mention anyone specifically.

CHAPTER 1

Introduction

Our perception of the world, as humans, is inherently dependent on the input coming in through the different sensory modalities. The interpretation of such input into meaningful information can be multi-fold. Not only do we rely on combining the different forms of data received, but our brains are also able to filter out and focus on the most relevant parts of such input. As we endeavour to code such behaviour into computers, Deep Learning research has enabled us to make significant strides in the direction of scene understanding from videos which is a combination of visual and audio modalities. The goal of our work further constraints the problem to Action Recognition (AR) from egocentric videos with the help of vision and sound.

Action can be defined as an activity being carried out with a particular goal in mind. Considering such an understanding can be done in many ways, arguably the most intuitive form would be to look at a scene from a first-person *i.e.* egocentric perspective. This gives us a direct peek at the interactions of an agent with its immediate environment. Such applications already exist in areas such as service robotics where this could be helpful to recognize human actions and react accordingly [13]. Objects generate discriminative sounds through which they can be recognized, such as "a kettle boiling" or "frying potatoes". In its digital form, it has a sequential numerical representation depicting a temporal relationship. The combination of sound from specific objects with its changing property over time can help in our AR task. A small example would be "turn on" and "turn off" a tap. While the sound of running water would start in the first case, the latter can be classified by interpreting that the same sound stops after a while. In third-person videos, such data cannot be recorded if the action itself is happening at a distance. Previously, the audio was discarded while creating AR datasets because of its high cost of storage [7,

4] and the fact hardly any previous methods used it. Owing to the explosion of video recording and storage technology, it is much easier now to create such video datasets with sound included [10, 11, 27, 32, 35, 42].

In our work, we look at how audio contributes to the task of AR individually and in combination with other visual modalities. We also try to make use of the sequential temporal relationship in audio to find the most relevant chunks of it while looking at the visual frame. Research already does exist for combining of audio with visual modalities [39, 40, 44] but we go further into investigating how well such fusion works. We introduce a general framework to train AR models based on a mid-level fusion technique while including an attention layer to filter out the most important temporal sections in the sound. TBN [44] is a special case of our framework and we use it to create our baselines without attention alongside experimenting with various settings. Our contributions can be categorized as:

1. Studying audio effectiveness for AR as an individual mode
2. Studying compatibility of audio fusion with other visual modality for AR
3. Introducing various attention mechanisms to find the most relevant audio section while looking at a visual frame

Figure 1.1: Sample egocentric frames from Epic-Kitchens dataset [32]

CHAPTER 2

Fundamental Topics

2.1 Convolutional Neural Network

The field of visual learning from images has seen tremendous advances since the use of Deep Learning became more mainstream amidst the progress in hardware technology. The most common architecture for models related to this is the Convolutional Neural Network (CNN). Typically it consists of one or more convolution layers followed by Fully Connected layers. Most image recognition algorithms make use of feed-forward architecture where the data is forwarded from the input to the output in a single pass. Both input and output take the form of tensors and their shapes depend on the type of input. Generally, it will consist of the number of images, image height, image width and the number of channels. Though this is applicable for 2-dimensional CNNs for image processing, they can also be used to process other forms of 2-dimensional data like audio spectrograms. The earlier forms of CNNs like VGG [16] used multiple deeply stacked convolution and pooling layers to perform image recognition tasks. Resnet [17] improved upon it by introducing residual connections. Inception networks [19] reduced computation overload via inception layers. These networks still form the core of much of current research in classification tasks even though the performance of the vanilla versions of these algorithms have been far surpassed by now. Despite its usage in various scenarios, we restrict our discussion to classification based CNNs.

The typical architecture of a CNN involves the following layers:

2.1.1 Fully Connected Layer

A Fully Connected (FC) layer is a linear layer which connects every node or neuron to every node of the output. This method allows feature learning over the entire input and the architecture is supposed to emulate the neural connections in a human's brain. Mathematically it can be represented by a matrix multiplication as,

$$f(\vec{X}, \theta) = W.\vec{X} + b \tag{2.1}$$

where, X is a vector input of size D, W is the weight matrix *i.e.* the values for each node of the linear layer of dimensions $D \times O$ with O being the output dimension and b as a bias term that is added to the feature.

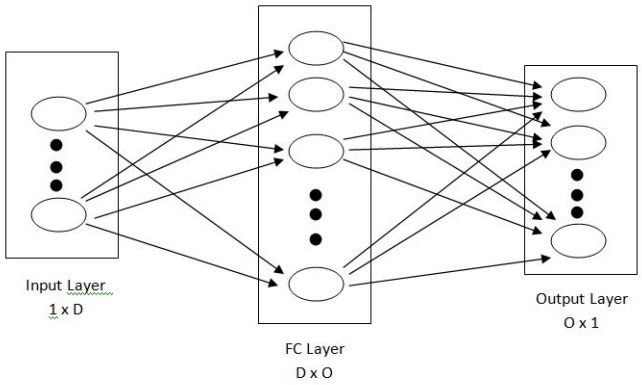


Figure 2.1: A single FC layer mapping an input to an output

In classification based CNNs, the final layers tend to be a collection of one or more FC layers. A softmax operation is utilized on the output of the the last layer to generate probabilities over the number of classes.

2.1.2 Convolutional Layer

A convolutional layer involves applies a convolution operation on an input image to generate feature maps. A sliding window approach computes the features over localized regions of the image. This makes the transformations more flexible and the computation efficient without much negative impact on spatial dependencies of the features. The convolutional kernels are defined as $height \times width \times channels$, where channels correspond to the number of feature maps.

2.1.3 Activation Layer

An activation layer is used to infuse some non-linearity to the generated feature maps. The most common form of activation function in a CNN is the Rectified Linear Unit (ReLU) which suppresses the negative values in the features by clipping them at 0. ReLU can be defined as,

$$f(x) = max(0, x) \tag{2.2}$$

where x is a value in the input feature map to the pooling layer.

2.1.4 Pooling Layer

The concept of pooling refers to the downsampling of features by using a consensus like average or max over localized regions. The most common form of this is maxpooling which selects the maximum value over a small kernel window and slides this over the feature map to create a smaller sized representation. There are however other forms of pooling like averaging and they can also be applied over the entire feature to form a global representation. Figure 2.2[1] shows a sample maxpooling operation over a 2-dimensional matrix with a 2×2 kernel and slide length of 2.

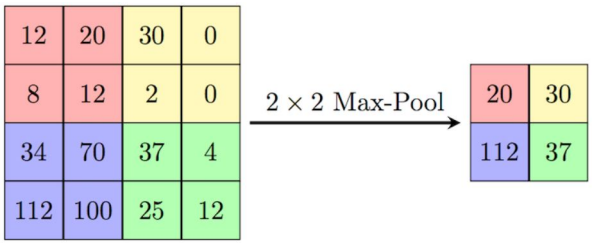


Figure 2.2: A sample Maxpooling operation

2.2 Optimization

Optimization in deep learning involves minimizing an objective function termed as loss. Gradient descent is a method where the network tries to find a minima by stepping along the negative gradient of the loss function (Equation 2.3). In Deep Learning, it is implemented by calculating the gradient of each layer, starting at the loss and traversing the entire network backwards via the chain rule of gradient calculation. This is called backpropagation. The gradient descent is defined as,

[1] https://computersciencewiki.org/index.php/Max-pooling_/_Pooling

$$L(\theta) = -\frac{1}{N}\sum_{i}^{N} f(\theta^T, x_i),$$
$$\theta_{t+1} = \theta_t - \eta \nabla L(\theta) \tag{2.3}$$

where, $L(\theta)$ is the loss function over the parameters θ, N is the size of the batch, t is the time step and η is a multiplication factor called learning rate which dictates the speed at which the function steps towards the minima.

Due to the difficulty of statistics computation over large datasets and processing them as a single input unit, the data is fed as small mini-batches into a deep neural network. The optimization happens iteratively over each of these mini-batches by penalizing misclassification for each iteration. This is known as Stochastic Gradient Descent (SGD) [3]. It is important to note that tuning the learning rate, as the loss converges towards optima, is helpful to prevent the optimizer from oscillating around the minima. SGD does not do this implicitly and there are other optimization algorithms to achieve it. We do not discuss them here as our experiments are entirely based on using SGD.

An additional constraint can rise with the lack of suitable hardware to fit large mini-batches. In such cases, it is prudent to utilize gradient aggregation schemes by summing the gradients over a fixed number of iterations k. The backpropagation is done over all k iterations after scaling the loss by the same. [23] shows that using too large mini-batches can cause optimization to become difficult. Equation 2.3 can therefore be modified as,

$$L(\theta) = -\frac{1}{kN}\sum_{i}^{N} f(\theta^T, x_i),$$
$$\theta_{t+1} = \theta_t - \frac{\eta}{k} \nabla L(\theta) \tag{2.4}$$

2.3 Audio Processing in Deep Learning

Audio in its digital form can be represented as a continuous sequence of numerical values corresponding to its waveform. The amount of information available in such a representation is represented by factors such as bit-rate, frequency and number of channels. Technically, extracting patterns from such single channel audio data is possible with networks using 1-dimensional convolutions or only FC layers. A lot of research such as that in [18, 33] work with raw audio to utilize the information available in it fully. However, it could be argued that for classification tasks, a more compressed representation like audio spectrograms work well enough as shown by [25]. Additionally, being a 2-dimensional representation, existing CNNs like VGG or Resnet can be used to perform classification using spectrograms without any significant modifications to the network.

In our experiments, we use the log-spectrogram representation obtained by applying a Short-time Fourier transform (STFT) [2] on an audio sample. STFT can be applied on any signal $x(n)$ at time n, by applying a window function $w(m)$ of length m on a short chunk of the data over which the Fourier transform is computed. The window is then slid along the length of the data and a piece is sampled at every fixed interval of size r. The STFT representation can be defined by,

$$\begin{aligned} X_m(\omega) &= \sum_{n=-\infty}^{n=\infty} x(n)w(n-mr)e^{-j\omega n} \\ X(\omega) &= \sum_{m=-\infty}^{m=\infty} X_m(\omega) \end{aligned} \tag{2.5}$$

where ω is the continuous signal, $X_m(\omega)$ is the Fourier transform over a data chunk and $X(\omega)$ is the discrete Fourier transform over the entire data sample. A single window transform is considered as a vertical line in the new representation, which represents the original signal as the phase and magnitude over time and frequency. The spectrogram is obtained by squaring over the magnitude of the STFT. However, since the range of perceptible frequencies to us as humans is small, the spectrogram values are converted to the log-scale. The log-spectrogram can be represented as,

$$\text{log\_spectrogram}\{X(\omega\} = \log(|X(\omega)|^2) \tag{2.6}$$

There are other forms of spectrogram representation like the log-mel spectrogram [5]. It works similar to STFT in using a sliding window to generate Fourier transforms but then converts the frequencies to the mel-scale. The usage of mel-scale [1] is recommended where only a short range of frequencies are necessary to be processed from the audio signal such as in Speech Recognition. This means the number of frequency bins in log-mel representation is lower and therefore it would be more sensitive to irrelevant sound such as noise. Hence, we decided to carry on our experiments with log-spectrograms calculated by STFT.

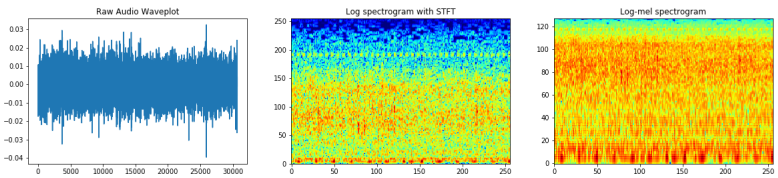


Figure 2.3: Audio Representations

CHAPTER 3

Related Work

The application of audio in Video Analytics (VA) research is fairly recent. In addition to our priority towards audio, we also study attention networks for their suitability to understand temporal dependencies. Therefore the related works with respect to our own can be classified as below:

1. Audio Learning Tasks
2. Action Recognition from Videos
3. Multi-modal Learning
4. Attention Networks

3.1 Audio Learning Tasks

Audio processing compared to that of images can be tricky because of its different forms of representation. In its raw form, it is a continuous sequence of numerical representations. However, the raw audio representation is of 1-dimensional form and most Deep Learning research for video classification has been focused towards 2-dimensional networks. Due to its variation in size arising from characteristics such as frequency, channels and bit-rate, most audio processing tasks use some form of compression to standardize this representation. This can take the form of mono-channel audio with normalization or conversion to log-spectrograms (Section 2.3), which is of 2-dimensional form. Research has also been done to learn generation of such 2-dimensional representations instead of hand-crafted spectrograms. [41, 53]

The Audio Learning tasks, related to our work, can be further sub-categorized as a) Sound Classification, [9, 25, 28, 37, 39, 45] b) Sound Localization, [31, 38] and c)Audio-Visual Representation Learning [21, 36, 52, 34]. Audio classification by [25] shows that audio spectrograms can be effective for classification with CNNs such as VGG [16], Resnet [17] and BN-Inception [19]. This is in close relation to our work as AR itself is a form of classification task [12]. Unsupervised representation learning between visual and audio samples was done by [21, 31] from unlabelled videos before being applied to classification. Such unsupervised representations might help in separation of relevant and irrelevant audio before attempting classification. The self-supervised approach of [36] via multisensory features attempts a similar task of representation learning. But it generate features from raw audio with 1-dimensional convolutions and a mid-level fusion with the 3-dimensional video features. In our research, we attempt to achieve a combination of the classification and audio-visual representation learning task while attending on the audio using a fully supervised approach.

3.2 Action Recognition from Videos

Till recently, AR has depended on using only visual modalities such as the RGB and Optical Flow frames. [14, 20] Partially this was due to the unavailability for audio in most datasets. 3-dimensional networks are useful in learning spatio-temporal relationships and hence prove useful in such tasks as AR [22, 30, 43, 50, 56]. The two-stream Inflated 3-dimensional CNN (I3D) proposed by [22] is a novel method to process sequential frames for the purpose of learning video features in both spatial and temporal scale. [50] addresses the problem of high computation cost in 3-dimensional networks by introducing channel-separated convolutions as a means to increase the efficiency of state-of-the-art 3-dimensional models used in Video Classification, such as I3D [22], SlowFast [43], 3-dimensional Resnet [24]. SlowFast[43] proposes a two-stream approach to process motion with both dense and sparse temporal sampling to capture finer temporal features alongside spatial semantics respectively. This allows the network to learn more comprehensive video features while comparing various kinds of action with differing speeds. Audio-visual SlowFast [56] adds upon this work by creating a fast pathway for audio sampling and using a multi-level fusion with the visual features.

The use of multi-stream networks with fusion is also common while using 2-dimensional networks. While multiple fusion techniques have been explored, Temporal Segment Network (TSN) [20] used a sparse temporal sampling to merge predicted softmax scores from a dual-modality network as a late-fusion. They were the first to introduce an aggregation over temporal scores after dividing a trimmed action clip into multiple segments. Temporal Binding Network (TBN) [44] utilizes a similar approach to split the action clip into

multiple segments but asynchronously samples data. However, their approach involves a mid-level fusion technique while using audio in addition to RGB and Optical Flow. This is closely related to our work as we use the same mid-level fusion. In comparison, we feed the network a larger audio sample and attend on it to let the network decide which is the most relevant audio part. We use the TBN approach to generate baselines for comparison.

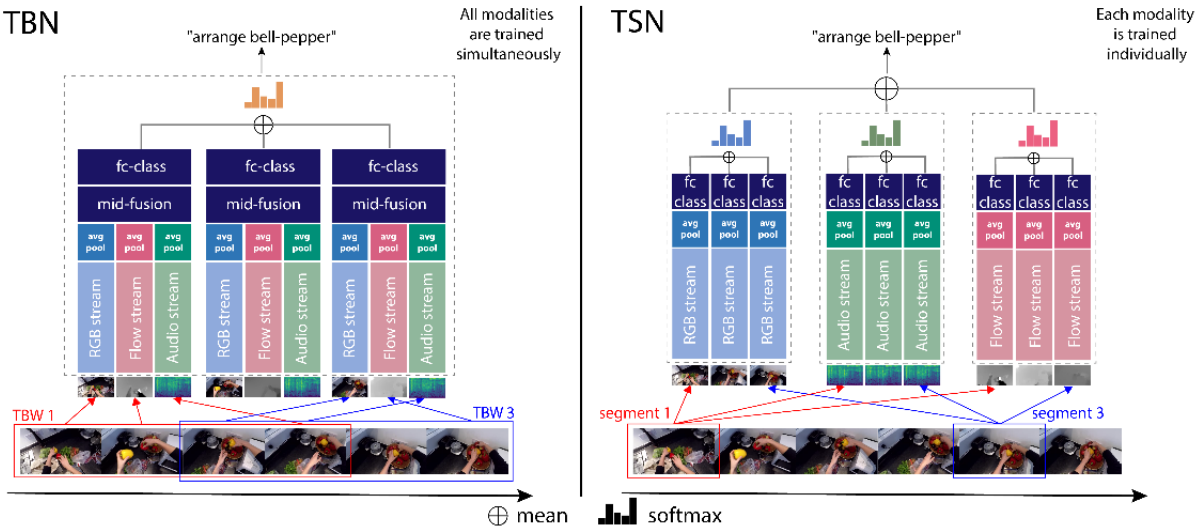


Figure 3.1: TBN v/s TSN architecture [44]

3.3 Multi-modal Learning

Multi-modal learning methods deal with combining information from varying sources and the feasibility of such fusion. Though such techniques [14, 34, 36, 40, 55] have shown promise in improving model performance, there are certain disadvantages associated with it. A comprehensive study of the single modal vs multi-modal training is done by [54] for classification tasks and the results exhibit the differing speeds at which different modalities overfit. Multi-modal networks tend to overfit faster due to their increased complexity. An interesting contribution from audio-visual SlowFast [56] is the inclusion of a dropout mechanism with the audio pathway, to mitigate the problem of the audio overfitting. We use this approach to evaluate if it works for our model with mid-level fusion.

3.4 Attention Networks

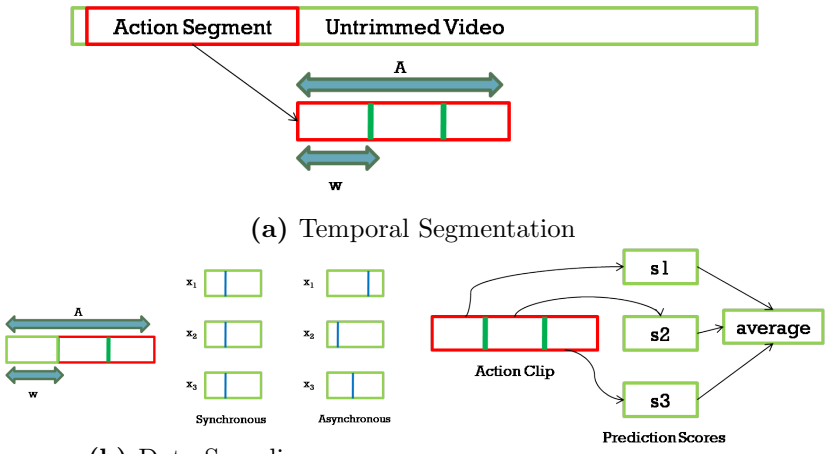
Attention based networks in Deep Learning refer to the ability of models in learning to focus on a specific area of input or features for better understanding of the task at hand. In the past few years, attention-based transformer networks [15, 29] has gained immense popularity in creating more robust models while focusing on the more pertinent sections of the data without additional supervision. Long-short term Attention proposed by [49] is a novel Recurrent Neural Network mechanism to generate attention over an entire video sequence while focusing on the spatial features as well. High-level feature aggregation by [48] creates high level features representations across video sequences with intermediary attention modules to focus on relevant local features. "Listen to Look" [51] recently unveiled a mechanism to learn a selection of the most interesting visual frames from an untrimmed video by looking at image-audio correspondence pairs. Despite being out of scope from our research, it lends credence to the fact that such cross-modal attention to learn temporal relations is possible. Multi-headed Attention [29] was developed for use in Natural Language Processing (NLP), to learn the sequential dependencies among words of a sentence. We extend this idea to learn similar sequential relationships that exist along the temporal dimension of a spectrogram. We argue that the temporal relationship in a spectrogram can still be found in the downsampled feature after feeding it through a CNN. It should be possible to learn this relationship by looking at the RGB feature to extract more robust audio features while using attention.

CHAPTER 4

Data

4.1 Epic Kitchens Dataset

Human-object interactions can be observed more comprehensively from an egocentric viewpoint. In this case, all activities are relevant with respect to the immediate environment including the action understanding *i.e.* our interactions with it. Egocentric videos also tend to have a richer and more relevant source of audio data than that in third-person videos. We train and test our models on the Epic-Kitchens dataset (currently known as Epic-55) [32], the largest collection of annotated egocentric videos, at the time of this project. A significant challenge of working with this dataset is its variation in settings. The videos were recorded in 32 kitchens of 4 cities at different times of the day. They are completely unscripted and were annotated from narrations provided by the recorders in multiple languages. The action is defined as a combination of Verb and Noun classes. There are 125 Verb and 352 Noun classes. Not all such combination are valid and only a meaningful combination of the Verb and Noun is considered an action. The dataset is split into a training set and two separate test sets, known as Seen and Unseen. The Seen set contains videos of kitchens already present in the training set while the Unseen set contains novel kitchens. Hence, it serves as a measure of comparing the "overfitting v/s generalization" problem prevalent in Machine/Deep Learning research. The labels are highly imbalanced throughout the dataset. If we consider unique combinations of a Verb and Noun in the training set to be an action, there are 2512 action classes in the training set, of which the top-10 most frequent classes have 4026 (14.14%) instances among a total of 28472. There are 2039 (81%) action classes which have less than 10 occurrences in the training set. In some cases, additional difficulties might also arise from different annotations of



(a) Temporal Segmentation

(b) Data Sampling

(c) Temporal Aggregation

Figure 4.1: Temporal Segmentation and Aggregation

the same class having high variance in the action boundary definitions along with invalid annotations. A look at the annotations of "put plate" shows it has 403 (1.42%) instances with a mean length of 1.667 seconds and variation of 5.319 seconds.

4.2 Temporal Segmentation and Aggregation

The sequential nature of video frames has the advantage of providing a variety of data samples to be chosen from the same annotation for classification. Each sample is processed individually to generate a class prediction. A consensus or aggregation over all the samples would, in turn, be considered as a single prediction over the entire annotated video clip. This form of temporal segmentation was introduced by TSN [20] and later used by TBN [44].

Let us consider an untrimmed video with an annotated action segment of length A. The trimmed action video is divided into N segments with equal widths (Figure 4.1a). Let there also be inputs as $X \epsilon \{x_1, x_2, \cdots x_m\}$ for $M \epsilon \{1, 2, \cdots, m\}$ modalities with frame rates of $\{r_1, r_2, \cdots r_m\}$ from each temporal segment. The data sampling in each segment across modalities can be done both synchronously and asynchronously (Figure 4.1b). Synchronicity can be achieved via approximation in case frame rates do not match between modalities as,

$$x_i = \frac{r_i}{r_j} * x_j \tag{4.1}$$

where, x_i is the sampled input of modality i, x_j is the sampled input of modality $j \; \forall \; i, j \; \epsilon \{1, 2, \cdots m\}$ and $i \neq j$.

This can be extended for any number of modalities and to achieve synchronicity we always select the first modality as the anchor to compare the frame rates. We can tune this sampling mechanism by changing the number of segments or randomizing the data selection within the temporal windows, which would lead to more robust training. Due to the unscripted nature of the videos, some instances also contain irrelevant background noise such as music. This could pose a challenge for us as our attention method does not try to filter unnecessary sounds if it overlaps with the relevant audio from the action.

CHAPTER 5

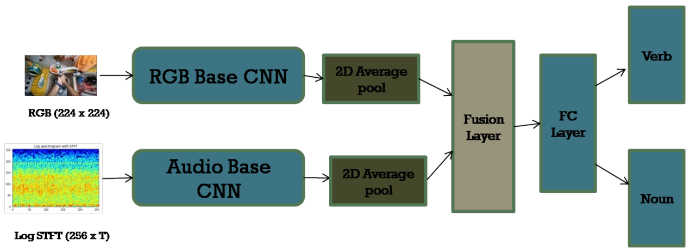
Model

5.1 Model Architecture

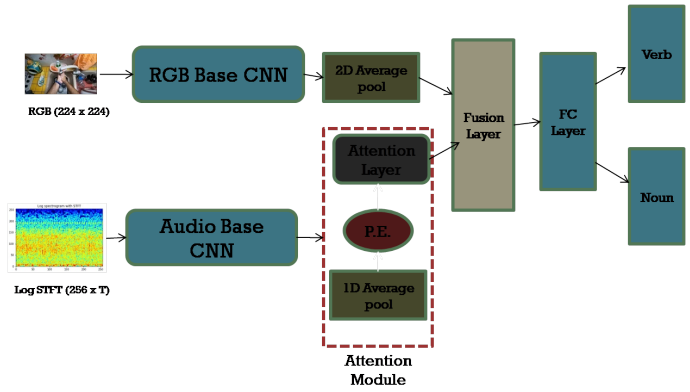
Our core model architecture (Figure 5.1) is a combination of pre-trained CNNs as feature extraction modules, followed by an attention layer to learn the interaction between audio and RGB features. At the end, we use FC layers for fusion and classification. Each modality has a base-CNN which generates a 2-dimensional feature. The visual modalities i.e. the RGB and Optical Flow frames are cropped to sizes of 224×224 and the audio spectrogram has a size of $256 \times T$, where T is dependent on the length of the audio clip. A global average pooling is used to compute 1-dimensional features over each modality which are then fused before being forwarded through the FC layers. The attention model replaces the pooling layer of the audio network with a 1-dimensional average pooling which takes the mean only along the vertical axis *i.e.* frequency dimension. We do this to preserve the temporal dependencies that exists in the downsampled audio feature.The classification scores of each class are averaged over the temporal segments to get the final prediction as,

$$\text{score} = \text{softmax}(\frac{1}{N}\sum^{N}(G(F_1, F_2, \cdots, F_m))) \tag{5.1}$$

where, $F_i = f(x_i)$ is the feature generation function for modality $i \ \forall\, i \,\epsilon\, \{1 \cdots m\}$, G is the fusion function applied over the concatenated features and N is the number of temporal segments.



(a) Model without attention

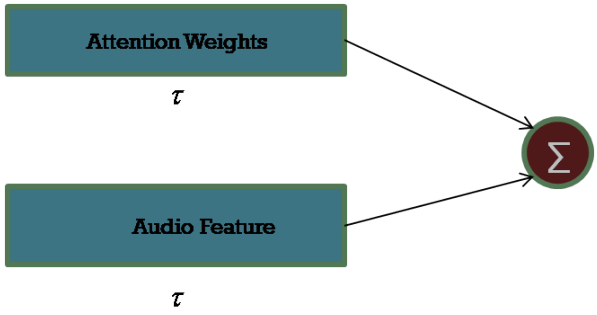


(b) Model with attention

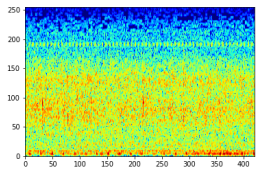
Figure 5.1: Model Architectures (FC Layer = Linear Layer + ReLU + Dropout)

5.1.1 Attention Mechanisms

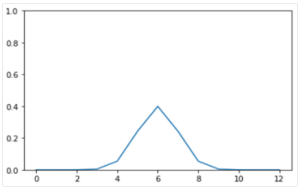
The necessity to suppress irrelevant data to better understand a specific task like AR is without question. In case of Deep Learning, this would enable models to generalize more by filtering out noisy sections of data. We base our attention mechanism on this argument and assume the sequential relationships along the temporal dimension of the audio feature can be used to calculate a more robust feature compared to averaging along the same. We make use of different forms of attention to generate a probability distribution, termed as attention weights. Let us consider, an audio feature with length τ along the temporal axis. A single unit along this axis can be assumed to correspond to a small window in the original spectrogram input with horizontal length T. A sum over dot product between the audio feature and attention weights of length τ would result in a 1-dimensional representation (Figure 5.2a). In line with that, we propose different forms of attention starting with preset beliefs called Fixed Attention followed by learned attentions namely Multi-headed Attention, Unimodal Attention and Prototype Attention.



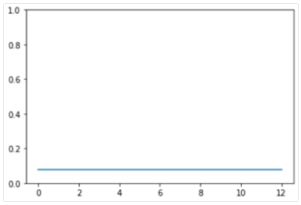
(a) Fixed Attention Layer



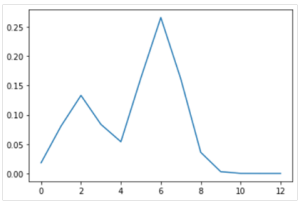
(b) Sample Spectrogram



(c) Fixed Gaussian



(d) Uniform Weights



(e) Fixed Loudest (Top-3)

Figure 5.2: Fixed Attention Mechanisms

Fixed Attention

The fixed attention layer chooses a preset probability distribution as the attention weights and applies it to all generated audio feature samples. They can be initialized to be any values like a Gaussian, uniform or others as long as they sum to 1. An intuitive form of attention could be to look at the loudest sections of the audio since the sound generated by the object is directly related to the action. A Gaussian distribution is created with its mean at the location of the loudest audio or highest amplitude in the spectrogram. We can fit multiple Gaussians to top-n highest frequencies and then take the mean over them to create the distribution. In the example from Figure 5.2e, the first and third loudest sections were approximately in the same location at the centre causing the peak on the left to be lower.

Multi-headed Attention

The self-attention mechanism by [29] is useful to learn the relation between a set of query and key-value pairs. The key-value pairs are a set of vectors with sequential dependency. In fact, the same vector is used for both. The temporal axis *i.e.* columns of a spectrogram can be said to possess a similar relationship. We attempt to attend on this audio feature by looking at the RGB one. The RGB feature vector is taken to be the query (Q) and compared against all the features from audio, the key (K) and value(V), using a scaled-dot product (Equation 5.2). The scaling factor ensures large dimensions do not have a negative impact on the gradient calculation during backpropagation because of their high variance. A softmax over this dot product generates the attention distribution which in turn is used to compute the weighted sum over the audio feature as,

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{5.2}$$

where, Q is the query with size $1 \times d_q$, K refers to the keys with dimension $\tau \times d_k$, V are the values with dimension $\tau \times d_v$, $d_q = d_k = d_v$ and τ is the length of the horizontal axis of the audio representation.

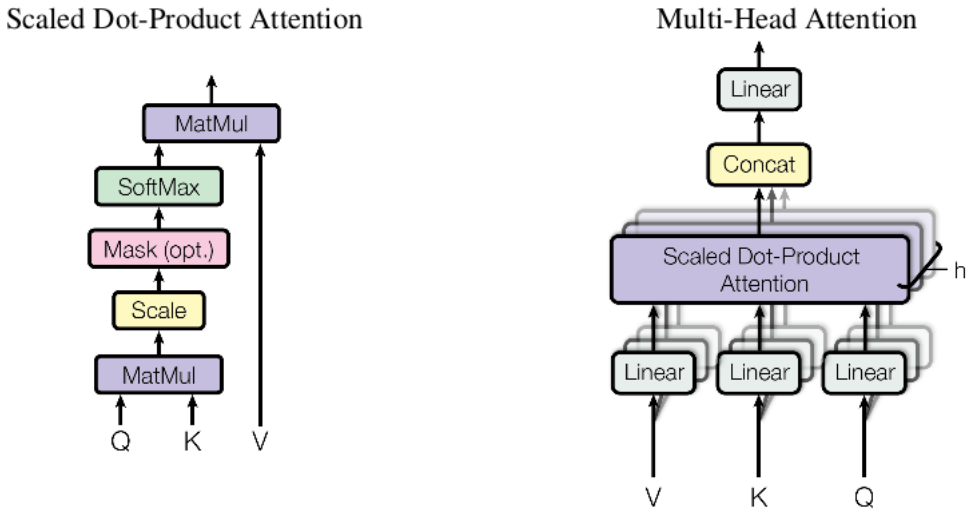
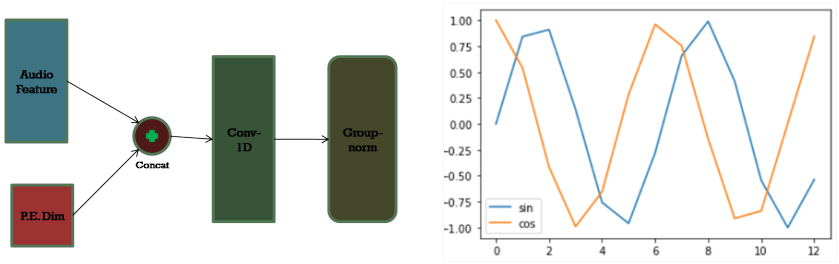


Figure 5.3: Multi-headed Attention (MHA) module [29]

Equation 5.2, however, refers to attention by projecting the query to a single location. A multi-headed feature means initializing multiple heads for the attention module to focus on different representation subspaces. This should discourage dominance in a single location. Multiple sets of weights for query and key-values pairs are initialized and trained in parallel to create different mapping representations as output. The different outputs from each



(a) PE Architecture

(b) Sine-Cosine Transforms

Figure 5.4: Positional Encoding (PE)

head are passed through a FC layer to obtain the final logits. A softmax over the logits gives us the probability distribution over the temporal axis. This can be represented by,

$$\text{MultiHead}(Q, K, V) = (head_1 \bigoplus head_2 \bigoplus \cdots head_h) \times W^O \tag{5.3}$$

where, h is the number of heads, $head_i$ is a single attention head *i.e.* $\text{Attention}(Q_i, K_i, V_i)\ \forall\ i \epsilon \{1, 2, \cdots, h\}$ and W^O is the weight matrix with dimension $h \times 1$.

The disadvantage of a naive implementation of Multi-headed Attention (MHA) is that it has no positional information encoded. We mitigate this by concatenating $2 \times d$ extra dimensions to the audio representation. Every alternate dimension in this is formulated as a sine and cosine transform with differing wavelengths of the form,

$$\begin{aligned} \text{Positional\_Encoding}_{pos,i} &= \sin(pos, i), \\ \text{Positional\_Encoding}_{pos,i+1} &= \cos(pos, i) \end{aligned} \tag{5.4}$$

where $i\ \epsilon\ \{1, 2, \cdots, d\}$ is the index and pos is the position.

The dimensionality of the key or value therefore changes to $d_k + d$ and is regained by passing through a FC layer. [29] terms this as Positional Encoding (PE) and it enables the Multi-headed Attention mechanism to take this information into account while learning the attention (Figure 5.4).

Unimodal Attention

The MHA mechanism in Section 5.1.1 interacts between the query and key-value pairs *i.e.* RGB and audio features respectively, to find pertinent sections to attend to. We propose an alternate attention mechanism by looking only at the RGB features and term this as Unimodal Attention (UA) (Figure 5.6). The 1-dimensional RGB feature representation is fed to a FC layer to generate logits of dimension τ. We consider these logits as a categorical distribution and apply a function over them to transform the logits into values that sum

to 1. The function can be any that can convert the logits to a probability distribution but also allows backpropagation through it. The Gumbel-Softmax trick [26] is a viable option for training such a network due to its ability to backpropagate while selecting the data according to a categorical distribution (Equation 5.5). The onehot mechanism of Gumbel-Softmax would mean only a single column of the audio feature is selected as all others are multiplied by zero. During inference, a softmax is applied over the logits instead, before computing the weighted sum, to keep the sampling fixed and eliminate the noise added by Gumbel-Softmax. We do not use the PE for UA as it makes no comparison between the positions in the input feature. The Gumbel-Softmax representation is defined as,

$$\text{Attention(RGB, Audio)} = \text{onehot}(\text{softmax}(\frac{g_i + \log(\pi_i)}{t}) \tag{5.5}$$

where, z is the multiplication term *i.e.* attention weight, π_i is the probability or logit score, g_i is i-th sample drawn from a i.i.d. sequence of $Gumbel(0, 1)$, and t is the temperature.

The Gumbel distribution term $Gumbel(0, 1)$ can be perceived as additive noise and may be defined as,

$$G(U) = -\log(-\log(U)) \tag{5.6}$$

where, G is the Gumbel distribution and U is a uniform distribution between 0 and 1.

The temperature in equation 5.6 acts as a smoothness factor and results in a uniform distribution as its value is increased (Figure 5.5).

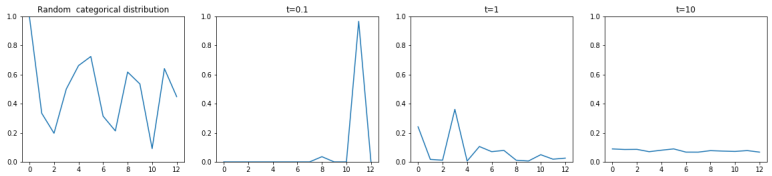


Figure 5.5: Effect of Gumbel-Softmax temperature(t) on a random categorical distribution

Prototype Attention

The Prototype Attention (PA) mechanism is aimed towards making the attention training more robust by providing a set of different attentions for each data sample to select from. A group of preset distributions are initialized as the said prototypes in the form of a matrix, where each row is a prototype. The RGB feature is forwarded through a FC layer to generate logits as a distribution over

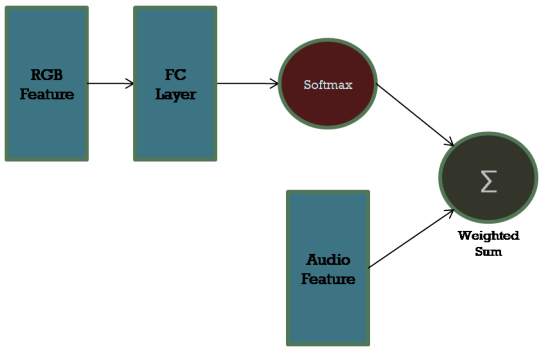


Figure 5.6: Unimodal Attention Architecture (FC Layer = Linear Layer + ReLU)

the number of prototypes. Once again, we make use of the Gumbel-Softmax trick to select an attention prototype from a predicted categorical distribution during training and softmax during inference. A matrix multiplication between the prototypes and probability values is then used to calculate the weighted sum over the audio feature representation. The backpropagation will carry the misclassification penalty back to the wrong selection of prototype and hence make it less likely to be selected for a specific sample. In theory, it should be possible to make these weights trainable, but we set them to be fixed attentions (Figures 5.7 and 5.8). The attention mechanism can be represented by,

$$\text{Attention(RGB, Audio)} = \sum_{\tau}(\sum_{p}(E(F_{RGB}).W^{PA}) \odot F_{audio}) \tag{5.7}$$

where, W^{PA} is the PA matrix with size $p \times \tau$, p is the number of prototypes, F_{RGB} is the RGB feature with dimension d_f, E is the function that produces logits of size p and F_{audio} is the audio feature with dimension $\tau \times 1$.

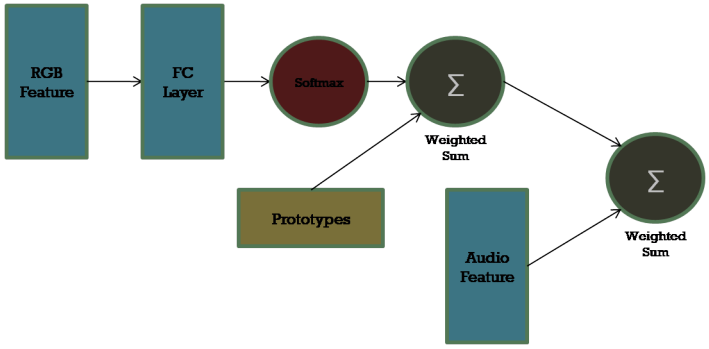


Figure 5.7: Prototype Attention Architecture (FC Layer = Linear Layer + ReLU)

We initialize our PA module with three fixed prototypes of Gaussian with variance σ as left-aligned, central and right-aligned respectively (Figure 5.8).

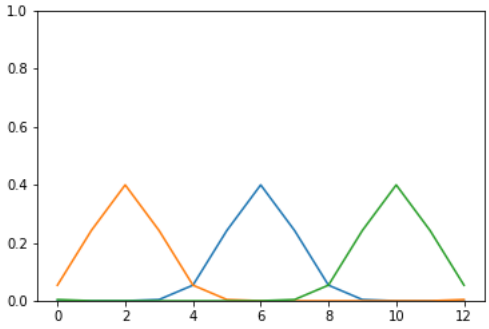


Figure 5.8: PA Samples with Gaussians with $\sigma = 1$

5.2 Loss Functions

The primary AR task of the network is trained by using a standard Cross-Entropy (CE) loss, defined as,

$$L_{CE} = -\sum_{x \,\epsilon\, X} p(x) \log q(x) \tag{5.8}$$

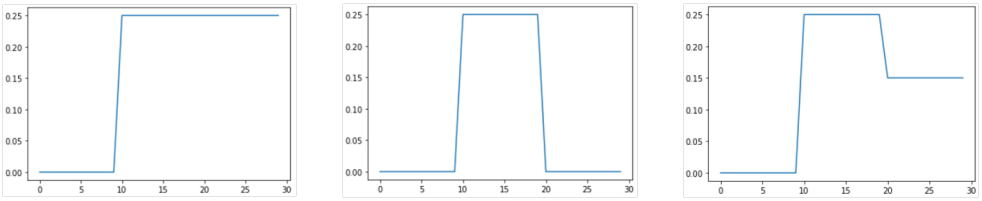
where, L_{CE} is the the Cross-Entropy loss, x is the input variable and $\{p, q\}$ are the two probability distributions over X to be compared.

While using an attention mechanism, the CE loss might be enough to drive the learning of classification task but it might not penalize the network fully if the learned attention tends towards some bias or any irrelevant patterns like noise in the audio. To mitigate this, we propose using an additional loss term called attention loss. A small decay factor is multiplied with the attention loss so the gradients are still dominated by the classification loss. It can be represented as,

$$L_{total} = L_{CE} + \gamma \cdot L_A \tag{5.9}$$

where, L_A is the attention loss and γ is the decay over attention loss.

We use three different losses namely a) Prior Loss, b) Contrast Loss and c) Entropy Loss. It was also observed that the contrast in the learned attention without using attention loss, tends to be very low. Hence, the primary goal of b) and c) are to increase the contrast in such a distribution. The randomly initialized attentions could possess bias of its own. In order to prevent b) and c) from causing the attentions to peak towards these biases, we also propose a stepping mechanism where the network initially learns the attention



(a) S Stepping **(b)** Inverse-U Stepping **(c)** ab Stepping

Figure 5.9: Stepping Mechanisms (Decay Rate v/s Epochs) for Attention Loss

without any additional loss and then the decayed attention loss is added to Cross-Entropy. The stepping mechanisms, illustrated in Figure 5.9, are plotted as decay rate v/s epochs. Each of them starts with a decay rate of 0 and is then set to a certain value after some epochs. Additional thresholding can be slotted in with any of the stepping mechanisms to fix the decay rate at 0 once the attention loss goes below a certain threshold. This would help the model focus more on the classification *i.e.* CE loss once the attention loss reaches a sufficiently low value.

5.2.1 Prior Loss

For Action Recognition, it is not premature to assume that the synchronous data from different modalities provide enough discriminative information for classification. This has also been shown by [31, 36] where the emphasis has been put on learning synchronicity between visual and audio data in an unsupervised manner. In line with that, we assume a prior over the distribution of the attention weights as a normal distribution with a variance σ. The comparison between the prior and the predicted attention can be done by any function to compare two probability distribution. We try out KL-divergence, Mean-squared-error and smooth-L1 for the same. The prior loss is defined as,

$$L_{prior} = \theta(a, b) \tag{5.10}$$

where, θ is the the loss function, a is the the predicted attention and b is the prior distribution.

The prior can be replaced by any distribution that one deems fit. Therefore, it is flexible based on the task or length of the sequence at hand.

5.2.2 Contrast Loss

The spatio-temporal attention mechanism of [46] for video classification uses a binary attention mask for contrast loss computation to put more weight on the relevant temporal regions. We utilize the same formulation to drive up the

contrast in the attention weights. A binary mask B (Figure 5.10) is computed by thresholding the predicted attention distribution U and the contrast loss, $L_{contrast}$ is defined by,

$$L_{contrast} = \sum^{\tau}(U \odot (1 - B) + U \odot B) \tag{5.11}$$

B is the mask generated by an indicator function with a threshold β (Equation 5.12) as,

$$B = \mathbb{1}M_i > \beta \tag{5.12}$$

where, M_i is the mask value at location i.

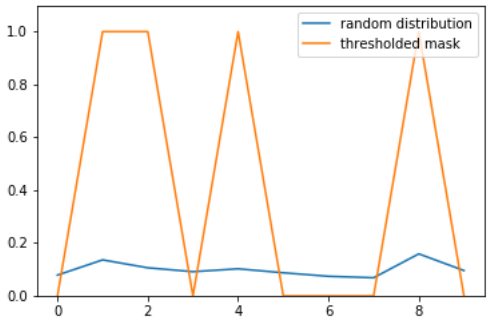


Figure 5.10: Binary mask with threshold of 0.1 over a random probability distribution

5.2.3 Entropy Loss

Entropy can be defined as the measure of uncertainty in a probability distribution [6]. A uniform distribution has a higher entropy value than that of a contrasting one. Therefore calculating the entropy of the predicted attention and attempting to minimize this automatically causes the attention to suppressing irrelevant areas. The entropy loss acts as an automatic regularizer by causing the learning to slow down when applied, which helps with the over-fitting problem in multi-modal learning. Since optimization techniques such as Stochastic Gradient Descent strive to lower the loss value and the original form of entropy is negative, the entropy loss is modified so that it is always positive. The entropy term is defined by,

$$I[a] = \sum_{a}^{A} p(a) \log_2 p(a) \tag{5.13}$$

where, a is the random event over predicted attention A and $p(a)$ is the probability of a occurring.

5.3 Relation to Temporal Binding Network

Our Attention-based Temporal Binding Network (ATBN) model is similar to the Temporal Binding Network (TBN) [44] architecture in applying a mid-level fusion to create a unified feature representation over various modalities before aggregating the prediction score over multiple temporal segments. Additionally, the authors of [44] extended the temporal segmentation of [20], to propse a Temporal Binding Window (TBW) approach in which data sampling can be done asynchronously. A small binding window is chosen with respect to the sampled data over the first modality. The data from remaining modalities are then sampled from within this binding window. In contrast, we pre-define the segmentation over the action clip and then select a larger audio clip, compared to that of TBW. The audio feature vector is generated by attending along the horizontal axis of the audio feature output from the base-CNN. TBN itself is a special case of our framework as we experiment over various settings to evaluate the viability of audio combination with other modalities both with and without attention mechanisms.

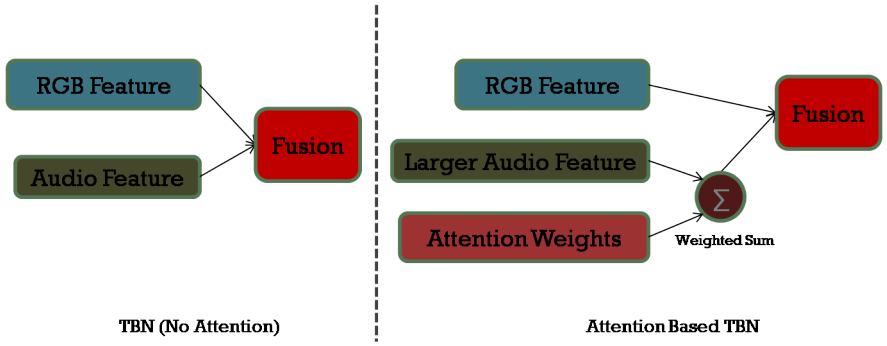


Figure 5.11: Comparison of TBN v/s ATBN Fusion Architectures

CHAPTER 6

Implementation

We implemented our Attention-based Temporal Binding Network with Python 3.7 and Pytorch 1.4 [47]. We acknowledge the base network architecture and data sampling was inspired by https://github.com/ekazakos/temporal-binding-network. A single Nvidia-RTX GPU of size 24 GB was used in our experiments. Our framework includes a Hydra [1] based config module which allows customizing the model hyperparameters and data sampling settings in a wide manner.

6.1 Data Sampling

The data is sampled as mentioned in Section 6.1 with each annotated action segment divided into equal length of N segments. During training, we use 3 segments and for test/validation, we use 25. The training data is selected randomly from within each temporal window across modalities. In the synchronous case, the index from the first modality is chosen as an anchor for the rest while using multiple modalities. By default, we only select the data from the central index during validation/testing. The main focus lies in attending on audio while looking at the RGB frame. Hence, we randomly choose a RGB index and then take the audio sample as being centred on the same *i.e.* synchronously.

We use the pre-extracted RGB and Optical Flow data publicly available from [32]. The audio for each video was extracted at 24kHz with a single channel.

We feed the network three-channel RGB images, ten channel stacked Optical

[1] `https://hydra.cc/docs/intro/`

Flow frames and single-channel audio spectrograms as input. The Optical Flow is stacked in an interleaved manner over 10 frames. In both cases of visual modality, the shorter side of the frames is resized to 256 and then cropped to a square of size 224. During training, we use a multi-scale multi-crop method to generate more noisy samples, which aids in training. In validation/testing, a centre crop over the image is used. Standard data normalization is performed as well with the respective normalizing parameters of the pre-trained base-CNNs.

The audio log-spectrogram can be calculated using STFT. We use a "hann" window of length 512 with an overlap of 5 ms to calculate the spectrograms. The windows are zero-padded in case they fail to match the window length. There are 256 frequency bins and the length of the horizontal axis depends on the size of the audio clip. For 2.1 seconds of audio, the spectrogram size is 256×420. The audio feature size along the temporal axis is 13 while using BN-Inception as the base. Since it is more prudent to sample data with temporal alignment and in a synchronous fashion for AR, we attempt to strike a balance between choosing too large an audio sample and a too small one, for attention. In the end, audio clips of 2.1 seconds were used for the attention experiments. Action boundaries are exceeded in case the annotated segment is too short, otherwise zero-padding is used. The audio processing and spectrogram generation is done with librosa [2].

We create two separate validation sets to emulate the "Seen" and "Unseen" Kitchens evaluation scheme of the test set by [32]. A set of 14 videos, the same as [44], are held out for validation in the former case. The Unseen validation set does not contain any video from the training set, with a total of 81 videos. All training videos belonging to "P_25" and above were held out for the Unseen set.

6.2 Model

The base feature extraction model can be implemented using any CNN architecture and in our case, we primarily use the BN-Inception network. However, the framework will allow usage of different variants of the 2-dimensional Resnet and VGG networks. We only use the BN-Inception for our multi-modality experiments and also for the attention mechanism.

The base BN-Inception is initialized by pre-trained Imagenet weights [8] for RGB and audio. The Optical Flow network is initialized by the pre-trained Kinetics [27]. The Batch norms of each feature extraction module are all frozen similar to [20] except that of the first layer. The pooling layer for the attention-based audio module averages only over the frequency axis. The downsampled horizontal axis size of the audio feature was found to be 25 for a 4-second audio clip while using BN-Inception. We use this as an anchor to

[2] `https://librosa.org/librosa/`

calculate the axis size of audio clips with different lengths.

The additional FC layers for fusion and classification are initialized with weights sampled over a normal distribution with 0 mean and standard deviation of 0.001 along with 0 bias.

The PE layer in the attention module (Section 5.1.1) adds six extra dimensions to the extracted RGB features with varying periods of sine and cosine terms. The dropout threshold for the PE is set to 0.5. The PA module (Section 5.1.1) is initialized with three prototypes of Gaussians with a variance of 1.

6.3 Training

We use SGD (Section 2.2) to train the model with an initial learning rate of 0.01 over 30 epochs. The learning rate is decayed by a factor of 0.1 after 20 epochs. We use a standard batch size of 32 and the gradient accumulator method to backpropagate over every 4-th iteration during training. The gradients are clipped at 20 to mitigate the effect of over-fitting.

6.4 Inference

The inference is carried out over 25 segments of the annotated action clips (Section 6.1). We measure the performance by evaluating top-1 and top5 accuracy over Verb, Noun and Action. The Action accuracy is computed by considering cases to be true where both Verb and Noun have been correctly predicted. The metrics from both validation sets have to be considered in tandem for comparable performance. An improvement on Seen but not on Unseen is an indication of an overfit model. The top-1 accuracy is considered as the main evaluation metric though we calculate top-5 as well.

CHAPTER 7

Experiments

We carry out experiments by starting from individual modalities and working our way through multi-modal training without and with attention respectively.

7.1 Audio Experiments

In our first experiments, we examine if increasing audio lengths perform better. This batch of experiments uses only one temporal segment in both train and validation. We train the models by randomly sampling an audio clip from the whole trimmed action segment. The validation is done by taking an audio clip of equal length from the centre of the trimmed video clip. Table 7.1 shows that a larger audio clip performs consistently better in Seen case. There is a slight performance drop in Unseen for Verb with increasing length, but it improves for the Noun and Action. This is valid as audio by itself does not have any motion information but contains sounds generated by specific objects.

Table displays 7.2 that BN-Inception (BNI) performs better than the different Resnet variations. The "Flat Conv" experiment was carried out by replacing the first convolution layer of the CNN with two parallel convolutions of 3×1 and 1×3 kernels respectively. The feature maps are then concatenated before being passed through the rest of the network (Figure 7.1). The assumption is that learning patterns along the frequency and time axis separately can help in generating more discriminative features down the line. The results, however, do not show any significant improvement over the normal BN-Inception architecture. In Unseen, the Top-1 Verb metric of BN-Inception outperforms the flat convolution architecture by 4%. A point of warning here we would be that we fine-tune our hyperparameters on BN-Inception and use it across all models in Table 7.2. This might not be the best approach as

Dataset	Audio Length (Secs)	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	0.8	37.66	74.14	17.24	36.45	10.69	31.77
	1.28	40.54	74.81	18.16	38.66	11.65	34.15
	2.1	42.42	75.85	20.13	40.29	12.73	36.2
	4.0	**43.51**	**77.88**	**22.88**	**43.34**	**14.49**	**39.12**
Validation Unseen	0.8	**26.25**	**65.63**	7.56	20.43	2.99	16.35
	1.28	25.35	63.15	7.7	18.65	3.59	14.75
	2.1	25.09	60.49	8.5	19.88	3.85	15.32
	4.0	25.85	60.29	**9.74**	**22.06**	**4.11**	**16.63**

Table 7.1: Effect of audio length

different models tend to have their own hyperparameter sets on which they work best. Regardless, we do it for a fair comparison and keep our base CNN as BN-Inception through the rest of our work. The model comparison experiments used 3 segments for training and 25 segments for the validation, which is also kept constant through the rest of our experiments.

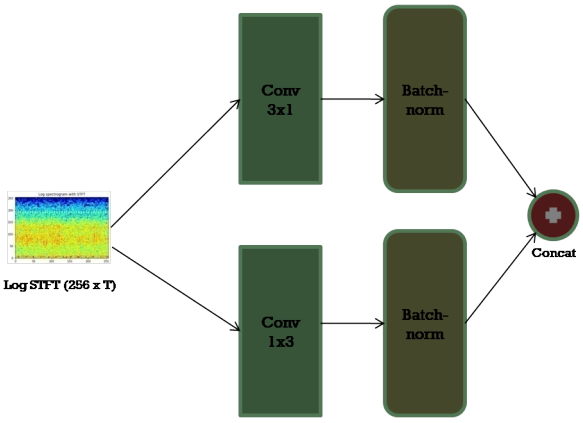


Figure 7.1: Flat Convolution architecture

7.2 Single Modality Experiments

The single modality experiments (Table 7.3) are aimed at observing the speed at which each of them train and the performance over each class of Verb and Noun. The hyperparameters for each experiment were kept the same except the number of epochs due to the varying speeds of overfitting. We chose an audio clip of 1.28 seconds for the single modality experiments. The RGB and

Dataset	Model	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	BNI	**43.97**	74.74	**21.49**	40.21	14.48	34.84
	BNI (Flat Conv)	43.28	73.75	20.95	39.19	**14.73**	**34.97**
	R101	38.32	73.76	18.66	37.2	11.9	32.82
	R50	40.17	75.31	20.17	**40.5**	12.61	34.95
	R34	40	**76.27**	19.33	38.16	13.19	34.11
Validation Unseen	BNI	**28.69**	58.22	**9.2**	20.75	**4.59**	15.78
	BNI (Flat Conv)	24.26	55.11	9.01	20.48	3.85	15.07
	R101	26.73	62.86	7.98	19.82	3.61	15.39
	R50	26.38	64.76	9.24	**22.2**	3.96	**17.44**
	R34	27.75	**65.78**	7.72	19.79	3.56	15.73

Table 7.2: Comparison of models

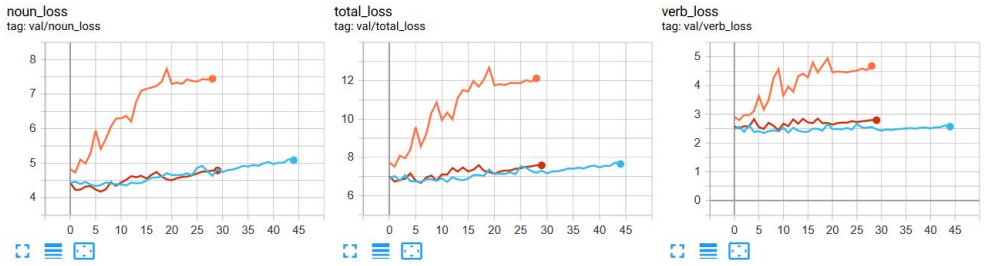
Audio modality were trained over 30 epochs with a learning rate decay after 20 epochs and the Optical Flow modality was trained over 45 epochs with a step down of learning rate at the 30-th epoch. The number of segments used for training and evaluation were 3 and 25 respectively. The choice of hyperparameters here was done to have a fair comparison with TBN [44] who train their models over 80 epochs. In our case, we found the network trains much faster and therefore had to adjust the epochs accordingly.

Dataset	Mode	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	RGB	48.85	**85.85**	**39.24**	**63.02**	**22.89**	**57.06**
	Flow	**55.93**	84.75	31.42	55.29	21.49	50.09
	Audio	43.97	74.74	21.49	40.21	14.48	34.84
Validation Unseen	RGB	35.05	76.97	**19.01**	**39.97**	8.72	**33.7**
	Flow	**44.27**	**78.4**	17.42	37.62	**11.02**	32.58
	Audio	28.69	58.22	9.2	20.75	4.59	15.78

Table 7.3: Single modality experiments

The results make it evident that visual modalities outperform audio in all aspects. Optical Flow is better suited towards Verb recognition while RGB is for Noun. This is comprehensive as motion information is encoded in the Optical Flow. The significant difference in Verb score between RGB

(a) Training Loss



(b) Validation Loss

Figure 7.2: Single modality Loss progression over epochs for Unseen validation set (Red = RGB, Blue = Flow, Orange = Audio)

and Optical Flow cause the Action accuracy to be higher for Optical Flow in some cases but in general RGB by itself contains enough information to discriminate between actions. The loss progression graphs in Figure 7.2 of each modality show that audio overfits extremely fast compared to the other two modalities. The "total loss" is the sum of the Verb and Noun loss over which backpropagation takes place.

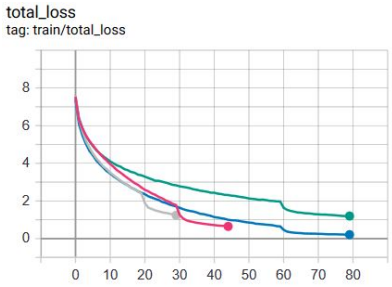
7.3 Multi-modality Experiments without Attention

The next round of experiments (Table 7.4) was done by combining modalities to test their suitability of fusion with each other. Once again, for a fair comparison, we tune the number of epochs and learning rate decay according to the overfitting characteristics of each combination. The RGB and Flow combination takes the longest. So we train it over 80 epochs. The Flow and audio fusion are trained over 45 epochs and the RGB with audio over 30 epochs. The audio length is kept to be 1.28 seconds for the experiments in Tables 7.4, 7.5 and 7.6.

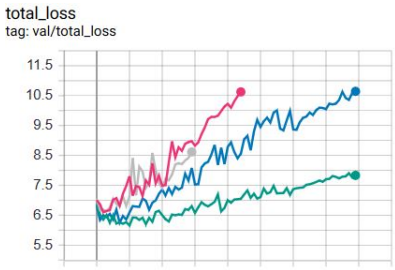
The results display that fusing audio with RGB gives a boost in accuracy scores in Seen set, but this is not as evident in the Unseen set. It is clear that this is caused by the overfitting effect of audio and thus holds back the model from performing better. RGB and Flow fusion in comparison performs well but throwing in audio with Flow proves even more problematic. The reason

Dataset	Mode	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	R+F	57.22	86.88	41.67	65.02	30.13	60.48
	R+A	56.73	85.25	40.5	65.47	28.33	59.18
	F+A	55.36	82.64	30.67	52.38	22.38	48
	R+F+A	57.49	85.54	40.8	64.89	28.76	58.6
Validation Unseen	R+F	45.6	79.54	20.29	43.09	12.14	37.17
	R+A	35.53	68.54	17.99	38.78	7.98	29.61
	F+A	38.97	68.68	13.66	29.26	7.06	23.48
	R+F+A	42.2	74.71	19.53	38.86	10.57	32.01

Table 7.4: Multi-modality experiments without attention (R = RGB, F = Flow, A = Audio)



(a) Training Loss v/s Epochs



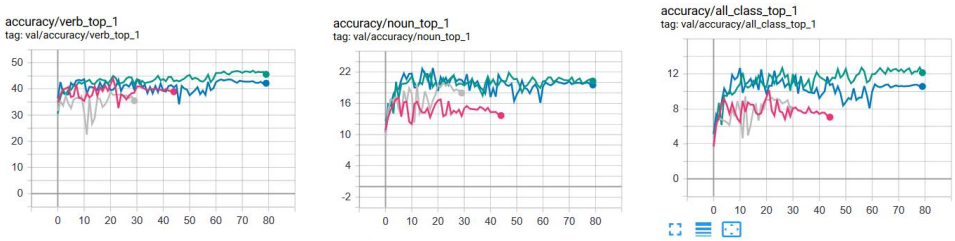
(b) Validation Loss v/s Epochs

Figure 7.3: Multi-modality total loss progression over epochs for Unseen validation set (Green=RGB+Flow, Gray = RGB+Audio, Pink = Flow+Audio, Blue = RGB+Flow+Audio)

for this is the highly contrasting training speeds of the two (Figure 7.3). The combination of all modalities therefore suffers and performs worse than the RGB+Flow fusion.

Henceforth, the epochs of all experiments are standardized as 30 with a learning rate decay after 20 epochs. Table 7.5 compares the effect of increasing the number of temporal segments during training with all modalities. We find 3 to be a good balance as a larger number shows a drop in performance in Unseen by $1-2\%$. The training with asynchronous data sampling over all modalities proves to be more effective as evident from Table 7.6. We compare the fusion of RGB and audio with lengths 1.28 and 2.1 seconds since we aim to use a larger audio clip for the attention experiments later (Table 7.7). The top-1 accuracy figures lie approximately within $\pm 1\%$ in Seen and $\pm 2\%$ in Unseen. The lower scores in Unseen are indicative of faster overfitting with larger audio clips.

We attempt to counter the overfitting by introducing an audio dropout



(a) Verb **(b)** Noun **(c)** Action

Figure 7.4: Multi-modality Top-1 accuracy progression over epochs for Unseen validation set (Green = RGB+Flow, Gray = RGB+Audio, Pink = Flow+Audio, Blue = RGB+Flow+Audio)

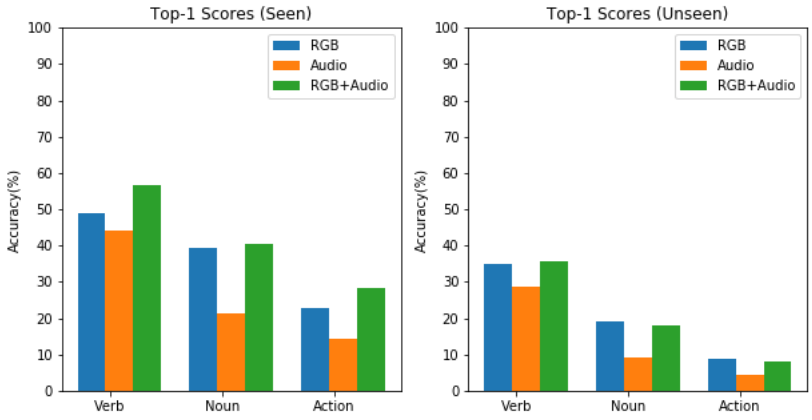


Figure 7.5: Comparison of top-1 accuracy scores between RGB and audio as individual and combined modalities

Dataset	No. of Segments	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	1	59.54	87.33	41.07	67.0	28.14	61.36
	3	60.42	**88.34**	42.53	67.08	30.65	**62.2**
	5	**60.84**	87.75	**44.59**	67.29	**32.48**	62.07
Validation Unseen	1	42.07	**79.17**	**21.95**	**46**	11.16	**39.04**
	3	**44.27**	76.99	21.85	44.02	**12.61**	36.83
	5	42.26	72.61	20.67	42.55	11.64	34.55

Table 7.5: Comparison of training all modalities with different number of segments

Dataset	Sampling Type	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	Sync	57.45	86.33	40.21	63.56	28.48	58.07
	Async	**60.42**	**88.34**	**42.53**	**67.08**	**30.65**	**62.2**
Validation Unseen	Sync	41.72	**77.04**	19.68	42.43	10.67	36.08
	Async	**44.27**	76.99	**21.85**	**44.02**	**12.61**	**36.83**

Table 7.6: Comparison of performance by models trained with synchronous (Sync) and asynchronous (Async) sampling

Dataset	Audio (seconds)	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	1.28	**56.45**	85.25	**40.5**	64.24	**27.43**	58.07
	2.1	55.62	**85.96**	40.33	**66**	26.47	**60.57**
Validation Unseen	1.28	**37.89**	71.42	**18.28**	**37.68**	**8.89**	**30.32**
	2.1	35.05	64.71	16.95	35.78	8.65	28.45

Table 7.7: Comparison of RGB+Audio fusion of audio clips with varying lengths

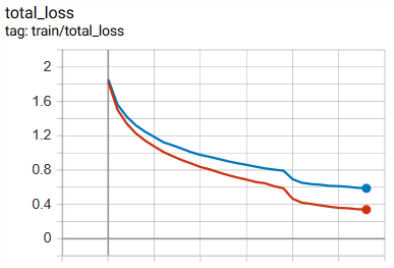
Dataset	Dropout Probability	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	0	60.42	88.34	42.53	67.08	30.65	62.2
	0.25	61.63	88.46	**44.79**	**69.55**	32.61	**64.75**
	0.5	**62.34**	**88.57**	44.66	68.25	**32.78**	63.2
Validation Unseen	0	44.27	76.99	21.85	44.02	12.61	36.83
	0.25	48.62	**81.97**	25.68	50.29	15.23	**44.23**
	0.5	**48.47**	81.14	**26.63**	**50.53**	**15.47**	44.05

Table 7.8: Effect of using audio dropout

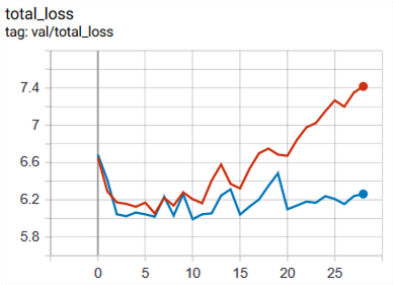
mechanism similar to audio-visual SlowFast [56] by randomly replacing the audio feature with zeros during training. The training was done with a fusion of all modalities and audio of 2.1 seconds as larger segments mean faster overfitting. Results from Table 7.8 and Figure 7.6 demonstrate the effectiveness of the audio dropout as we observe an improvement of $2-3\%$ in the Seen validation set and $4-5\%$ in Unseen validation set, across all classes.

7.4 Attention Experiments

In our next experiments, we move on to evaluations of the Attention-based Temporal Binding Network models. We start with the fixed attentions before



(a) Training Loss v/s Epochs



(b) Validation Loss v/s Epochs

Figure 7.6: Total loss progression without and with audio dropout (Red = No dropout, Blue = Dropout 0.5)

moving on to the three learned attention modules. Each of the learned attention modules are fine-tuned separately while keeping the previous hyperparameters from no-attention experiments fixed. Due to the detrimental effect of combining audio and Optical Flow, we only use RGB and audio for the initial attention experiments. Optical Flow is added to the best model obtained from the attention experiments to check if the overfitting persists. A Positional Encoding module is affixed to the beginning of every attention module but we also test its efficacy by removing it from some experiments. The audio length is kept at 2.1 seconds and we use synchronous sampling as the larger audio clip with attention should we equivalent or better than asynchronous sampling.

7.4.1 Fixed Attention

Attending on the loudest sections of audio permit the actions to be more distinguishable compared to other fixed attention experiments based on the results from Table 7.9. The fixed attention with uniform values is equivalent of averaging the feature over the time axis. The attention on the loudest audio, experiments show consistent improvement of $\pm 1\%$ over the uniform weights. The fixed Gaussian performs similarly except the top-1 Noun accuracy in Unseen where we see a drop of 1%. The attention on the two loudest audio bits has the most balanced metrics between Seen and Unseen. The two different experiments in Table 7.9 between top-3 loudest attentions are comparisons of synchronous and asynchronous sampling. The takeaway from the outcomes is that the two sampling mechanisms have similar metrics while attending on a larger audio clip. Overall, it proves that focusing on specific sections of the audio, even with fixed weights, can give a boost in the performance.

Dataset	Type	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	Gaussian	57.04	84.38	40.07	65.62	27.68	59.33
	Uniform	55.45	84.54	39.42	63.66	26.14	57.41
	Loudest 1	54.7	83.67	40.54	63.26	27.27	56.33
	Loudest 2	56.74	85.55	40.42	65.16	28.06	59.25
	Loudest 3	57.7	85.46	40.29	64.87	28.27	59.61
	Loudest 3*	**57.45**	**85.26**	**41.96**	**66.5**	**29.35**	**60.08**
Validation Unseen	Gaussian	37.54	71.78	16.39	37.63	8.27	30.54
	Uniform	36.21	68.65	17.4	36.92	7.98	29.21
	Loudest 1	37.69	71.82	18.55	39.42	**9.78**	**32.77**
	Loudest 2	**37.73**	72.19	**18.7**	**39.65**	8.83	32.42
	Loudest 3	37.46	**72.49**	17.77	37.92	9.12	31.61
	Loudest 3*	35.5	68.65	18.37	37.82	8.31	29.88

Table 7.9: Fixed Attention experiments ($*$ = asynchronous sampling during training)

7.4.2 Multi-headed Attention

Fixed attention experiments could have a limited effect at best. Therefore learning the attention weights would be more judicious. The MHA experiments are done by initializing 4 attention heads in parallel, for 2.1 seconds of audio, which should be large enough without compromising the accuracy (Table 7.10. In fact, most annotated action clips are smaller than 4 seconds. The PE definitively improves performance as evident from the Unseen scores in Table 7.11. However, training the MHA without additional supervision gives us little to no improvement over the baseline with the same audio length (Figure 7.7). The cause is that the learned attentions are mostly uniform and have very little contrast (Figure 7.8c).

Dataset	Audio (secs)	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	2.1	**56.94**	84.79	**40.84**	**63.99**	**27.89**	57.27
	4.0	55.99	**85.98**	40.13	63.91	26.51	**58.69**
Validation Unseen	2.1	**36.64**	**68.16**	**16.47**	**37.18**	**8.51**	**29.97**
	4.0	35.77	64.77	16.11	36.18	8.07	28.36

Table 7.10: MHA experiments with varying audio Lengths

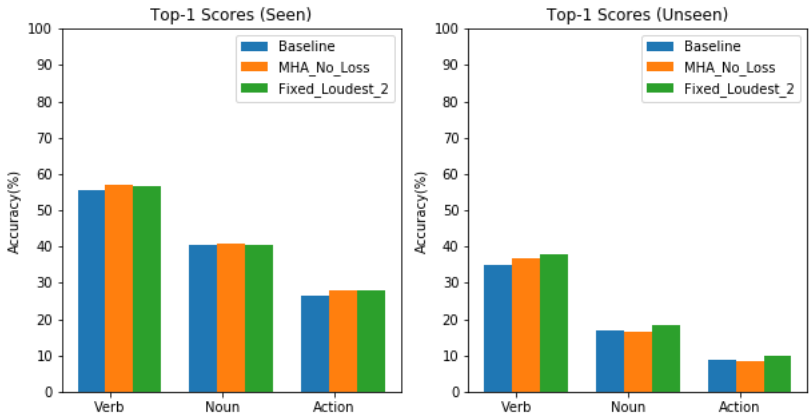


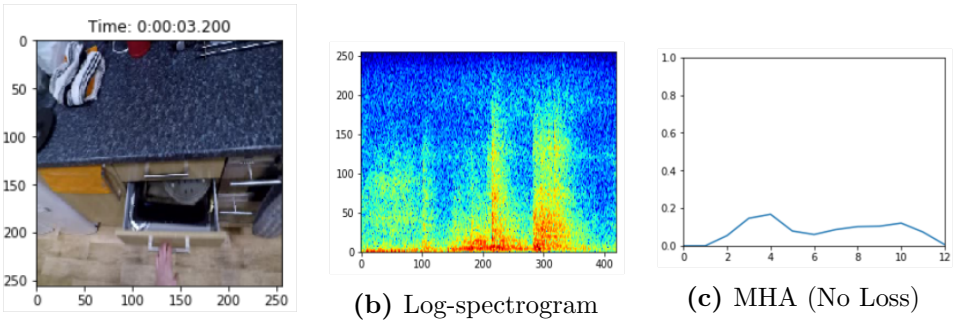
Figure 7.7: Comparison of top-1 accuracy scores between no attention, MHA and fixed attention models

Dataset	Type	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	no PE	**56.94**	**84.79**	**40.84**	63.99	**27.89**	57.27
	with PE	56.08	83.83	39.87	**64.7**	27.72	**58.32**
Validation Unseen	no PE	36.64	**68.16**	16.47	37.18	8.51	29.97
	with PE	**37.51**	66.92	**18.6**	**39.22**	**8.96**	**31.11**

Table 7.11: Effect of PE on MHA

The prior loss is meant to nudge the learned attention towards a prior belief of how the attention should be. Therefore, instead of using a fixed distribution, we model it slowly to a form resembling the prior. We use a Gaussian of variance 1 (Figure 5.2c) as the prior for experiments in Tables 7.12 and 7.13. The comparison among different forms of loss types and decay factors show KL-divergence to be the best choice with a decay of 0.25. Figure 7.8d already shows how the learned attention gets closer to resembling our choice of prior. We also observe that stepping does not have much effect when combined with the prior loss since we are only reshaping the learned attention towards a preset belief.

The learned weights from contrast loss (Figure 7.8e) display the problem of attention being skewed towards some bias. The attention in this case,



(a) RGB Image (b) Log-spectrogram (c) MHA (No Loss)

(d) MHA+Prior (e) MHA+Contrast (f) MHA+Entropy

Figure 7.8: Different learned attentions with MHA on a sample of "Open Drawer"

Dataset	Loss Type	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	KL	**58.83**	**85.92**	**41.25**	**66.12**	**28.6**	**60.07**
	MSE	56.58	85.75	39.25	64.9	27.38	59.71
	Smooth L1	56.38	85.44	38.58	65.32	26.88	59.69
Validation Unseen	KL	**38.71**	**70.62**	**17.61**	**39.74**	**9.77**	**32.53**
	MSE	36.55	68.48	16.61	35.27	8.55	28.55
	Smooth L1	35.75	66	16.6	35.99	8.86	28.93

Table 7.12: Comparison of different forms of Prior loss

though off centre, is not correct as it focuses towards an area where there is no sound from the drawer. The stepping is however very important here as, without it, this form of incorrect attention would be consistent throughout the dataset. The weights would then peak towards the bias of the randomized initial attentions. The two red peaks in the spectrogram are the place where the attention should be focused, which corresponds to the sound of the drawer being pulled out. We find the S-stepping method to add a decayed attention loss after 10 epochs, with a threshold of 0.1 and a decay factor of 0.25 to be

Dataset	Decay Factor	Step	Verb		Noun		Action	
			Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	0.5	N/A	55.07	**86.1**	**41.32**	64.37	27.64	59.25
	0.25	N/A	56.16	85.08	41	**65.08**	**27.72**	**59.4**
	0.25	10	**57.04**	85.22	40.21	64.33	27.39	58.25
Validation Unseen	0.5	N/A	36.56	70.81	17.21	38.28	8.95	31.38
	0.25	N/A	38.63	71.24	**18.72**	**39.89**	**9.34**	**32.75**
	0.25	10	**38.94**	**71.63**	17.92	39.15	8.65	32.07

Table 7.13: Comparison of decay factors and stepping with Prior loss

the best choice of hyperparameters of the contrast loss (Tables 7.14 and 7.15).

Dataset	Threshold	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	0.1	**58.75**	**86.54**	**42.51**	65.41	**30.61**	60.07
	0.2	56.53	85.87	40.79	**66.54**	27.81	**60.57**
Validation Unseen	0.1	**37.96**	**70.73**	**19.24**	**40.86**	**9.78**	**33.32**
	0.2	36.44	69.23	17.46	38.49	8.6	31.11

Table 7.14: Threshold comparison for Contrast loss experiments

Dataset	Decay Factor	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	0.25	**58.75**	**86.54**	**42.51**	65.41	**30.61**	**60.07**
	0.5	58.33	86.08	42.3	**65.85**	29.23	60.01
Validation Unseen	0.25	**37.96**	**70.73**	**19.24**	**40.86**	**9.78**	**33.32**
	0.5	36.35	70.11	17.97	38.99	8.58	31.8

Table 7.15: Comparison of decay factors for Contrast loss

In comparison to the Contrast loss, the Entropy loss appears more confident in its prediction with a central sharply shaped attention (Figure 7.8f). The performance metrics from Table 7.17 show how the different stepping mechanisms affect the learning. The U-stepping is simply the opposite of inverse-U and was done to prove the usefulness of stepping up the decay factor from 0. The results confirm our assumption as all other stepping mechanisms outperform it over all classes. The S-stepping and ab-stepping metrics turn out to be similar with a difference falling within $\pm 1\%$. A direct comparison of the figures from

S-stepping, inv-U stepping and ab-stepping would not be appropriate since the decay factors and step sizes are additional hyperparameters. They could be further tuned with respect to each method. We find S-stepping to be sufficient for our further experiments.

Dataset	Decay Factor	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	0.05	55.87	85.33	40.57	63.66	27.97	57.98
	0.15	56.78	86.75	41.92	65.71	28.35	60.61
	0.25	**59.08**	**87.25**	**42.74**	**66.42**	**29.69**	**61.32**
Validation Unseen	0.05	36.99	69.88	17.63	39.56	9.38	32.31
	0.15	37.97	70.92	18.1	38.26	8.93	31.49
	0.25	**40.39**	**75.91**	**19.77**	**41.77**	**10.66**	**35.9**

Table 7.16: Comparison of decay factors with Entropy loss

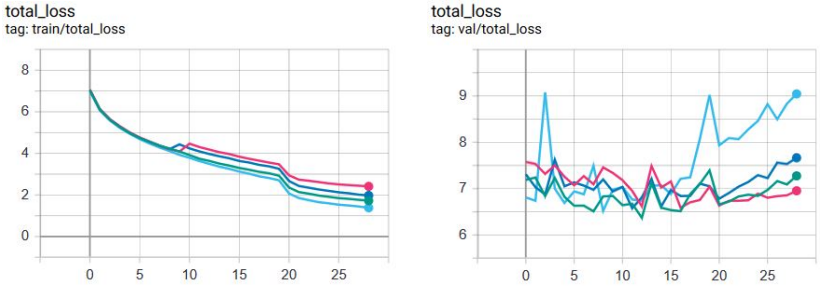
Dataset	Step Type (Epochs)	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	S(10)	59.08	87.25	42.74	66.42	29.69	61.32
	inv-U(10, 20)	58.87	86.96	43.37	67.28	30.61	62.24
	U(10, 20)	55.47	84.93	40.42	64.54	27.43	58.16
	ab(10, 20)	58.04	86.79	41.41	66.46	28.56	61.41
Validation Unseen	S(10)	40.39	75.91	19.77	41.77	10.66	35.9
	inv-U(10, 20)	38.66	73.1	19.34	42.24	9.84	34.87
	U(10, 20)	37.21	69.92	19.22	39.91	9.64	32.18
	ab(10, 20)	39.98	76.13	20.81	43.53	10.47	36.49

Table 7.17: Different Stepping mechanisms with Entropy loss

In the experiments from Table 7.18, we include a threshold of 0.2 with the entropy loss and S-stepping. We find that thresholding or a larger step size over epochs does not lead to any significant improvement over our previous experiment in Table 7.17. A more interesting observation was that the MHA with attention losses leads to automatic regularization of the overfitting effect from audio. The most prominent drop can be seen by the Entropy loss (Figure 7.9). The slight bulge in the training loss (Figure 7.9a) is the indication of the S-stepping taking effect. This is suggestive of the training being slowed down due to the additional noise brought by attention loss, hence restraining the model from overfitting.

Dataset	Step	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	10⋆	**59.08**	**87.25**	**42.74**	**66.42**	29.69	**61.32**
	10	57.87	86.25	42.8	66.12	**30.02**	60.95
	20	56.2	85.41	41.88	65.21	28.85	59.49
	25	56.87	84.58	41.09	64.83	28.27	58.48
Validation Unseen	10⋆	**40.39**	**75.91**	19.77	41.77	**10.66**	35.9
	10	39.46	75.82	20.27	**43.45**	10.19	**36.49**
	20	38.35	73.71	**20.43**	41.08	10.09	33.59
	25	36.59	70.85	19.21	38.92	9.21	31.09

Table 7.18: Comparison of different step sizes in epochs with Entropy loss and S-stepping (⋆ = no thresholding)



(a) Training Loss v/s Epochs **(b)** Validation Loss v/s Epochs

Figure 7.9: Comparison of total loss progression between no attention and MHA experiments (Light-Blue = Baseline without attention, Deep-Blue = MHA+Prior, Green = MHA+Contrast and Pink = MHA+Entropy)

7.4.3 Unimodal Attention

The Unimodal Attention (UA) uses the RGB feature as a reference to predict the attention weights and the Gumbel-Softmax trick to include some random noise while sampling the audio features. We compare the Gumbel-Softmax representation both with and without Onehot. The "No Onehot" representation simply removes the Onehot operation from Equation 5.5. The "Softmax" type in Table 7.20 replaces the Gumbel operation with a normal softmax over the logits. Since UA does not consider the relative positions in the audio features while including noise, we also compare its usage with and without PE. The final experiment, "with PE", includes the PE module while using the Gumbel-Softmax with Onehot representation. All the Gumbel-Softmax experiments outperform the normal softmax by $1-2\%$ in all classes for the Seen set. However, in Unseen, we do not find such a big difference except in

Verb. One can surmise, the randomized noise from Gumbel-Softmax slightly helps in boosting the score. The addition of Entropy Loss to UA does not prove effective. The cause is evident in Figure 7.10b. The UA learns a sharply contrasting attention weight using the same RGB and spectrogram combination from Figures 7.8a and 7.8b by itself, in comparison to the MHA with Entropy loss. Therefore it is more confident in its predictions and always attends on a single chunk of the audio feature. The addition of PE increases the Unseen scores across all classes marginally and therefore it can be chalked down to noise instead of any actual improvement. We also find that scaling the logit scores by any temperature value other than 1 can have a negative impact (Table 7.20).

Dataset	Type	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	Gumbel	57.06	**85.62**	**41.67**	**66.21**	**29.27**	**60.49**
	Gumbel (No Onehot)	56.2	83.66	40.84	63.87	28.48	56.78
	Softmax	55.45	84.33	39.08	62.49	27.01	56.35
	Gumbel + Entropy Loss	56.83	84.79	40.96	65.79	28.98	59.61
	with PE	**57.08**	84.34	40	63.74	27.22	57.58
Validation Unseen	Gumbel	36.95	71.97	18.6	40.51	9.27	32.95
	Gumbel (No Onehot)	37.37	71.38	19.27	40.56	9	32.33
	Softmax	35.56	69.9	19.13	38.98	9.02	30.74
	Gumbel + Entropy Loss	37.35	**73.01**	19.32	40.34	9.36	**33.1**
	with PE	**37.73**	71.45	**19.98**	**40.75**	**9.36**	32.74

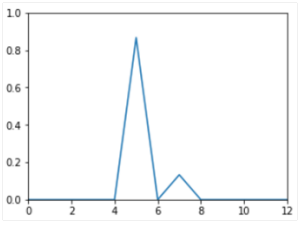
Table 7.19: Effect of Gumbel-Softmax, Softmax and PE on UA

7.4.4 Prototype Attention

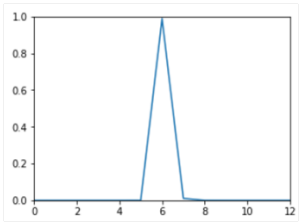
The prototype attentions are initialized with three Gaussians as central, left and right-aligned. The resulting weights show that the model always chooses the centrally aligned Gaussian for the attention (Figure 7.10b). This is indicative of the fact that audio-visual modalities in Action Recognition tend to be synchronized. Results in Table 7.21 do not point to much difference between Gaussian prototypes of differing variances. An important note here is, we do not explore further than these experiments at this point. Experimenting with other

Dataset	Temp	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	0.5	56.53	85.34	40.25	64.2	27.56	58.54
	1	**57.06**	**85.62**	**41.67**	**66.21**	**29.27**	**60.49**
	2	56.74	84.91	40.58	63.79	28.1	57.69
Validation Unseen	0.5	35.85	69.3	18.23	38.99	9.15	31.39
	1	**36.95**	**71.97**	**18.6**	**40.51**	9.27	**32.95**
	2	36.83	70.48	18.3	38.82	**9.41**	31.43

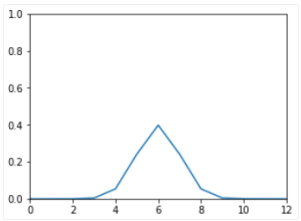
Table 7.20: Comparison of different temperatures of Gumbel-Softmax with UA



(a) MHA+Entropy Attention



(b) Unimodal Attention



(c) Prototype Attention

Figure 7.10: Comparison of learned attentions between MHA+Entropy, UA and PA

fixed prototypes such as loudest attention could return more comprehensive results.

Dataset	Variance	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	1	**54.78**	**83.54**	**41.13**	**63.52**	**27.72**	**56.81**
	2	54.45	83.05	40.75	63.37	27.47	55.95
Validation Unseen	1	35.61	**69.56**	**18.75**	38.25	**9.02**	**30.46**
	2	**36.33**	68.71	18.25	**38.35**	8.7	30.29

Table 7.21: Prototype Attention experiments with central, left and right aligned Gaussians as protoypes

7.5 Comparison

A comparison of the different MHA experiments shows that MHA with Entropy loss has the best performance. A further comparison with the current UA and PA experiments reinforces that observation ((Table 7.22)).

Dataset	Attention Type	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	None	55.62	85.96	40.33	66	26.47	60.57
	MHA (No Loss)	56.37	84.33	39.92	62.57	27.27	56.19
	MHA+Prior	57.04	85.22	40.21	64.33	27.39	58.25
	MHA+Contrast	58.05	86.54	42.51	65.41	**30.61**	60.07
	MHA+Entropy	**59.08**	**87.25**	**42.74**	**66.42**	29.69	**61.32**
	Unimodal	57.06	85.62	41.67	66.21	29.27	60.49
	Prototype	56.62	84.8	40.25	64.83	27.72	58.62
Validation Unseen	None	35.05	64.71	16.95	35.72	8.65	28.45
	MHA (No Loss)	36.64	68.16	16.47	37.18	8.51	29.97
	MHA+Prior	38.94	71.63	17.92	39.15	8.65	32.07
	MHA+Contrast	37.96	70.73	19.24	40.86	9.78	33.32
	MHA+Entropy	**40.39**	**75.91**	**19.77**	**41.77**	**10.66**	**35.9**
	Unimodal	36.95	71.97	18.6	40.51	9.27	32.95
	Prototype	38.32	72.1	17.99	38.35	9.03	31.52

Table 7.22: Comparison of MHA, UA and PA experiments

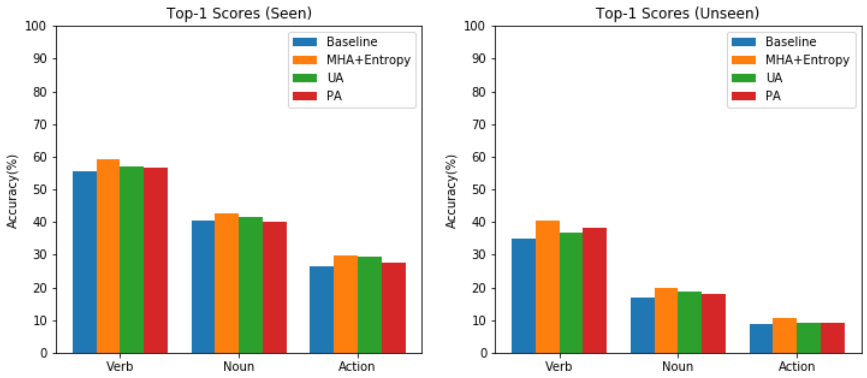


Figure 7.11: Comparison of top-1 accuracy scores between no attention baseline, MHA+Entropy, UA and PA

As mentioned previously, we add Optical Flow as an extra modality, to the

best performing model and compare it against the all modality baseline. In the all modality experiments as well, the audio is attended upon while looking at only the RGB feature (Table 7.23). The baseline is trained with an audio clip of 1.28 seconds and the data sampling is done asynchronously within the temporal windows. The attention experiments have audio clips of length 2.1 seconds. Only the RGB modality is sampled randomly and the others are sampled in sync with it. The all modality scores being greater than the fusion of RGB with audio, we can conclude that adding Flow to the mix, definitely improves the scores. The previously discussed problem of contrasting learning speeds between Optical Flow and audio (Section 7.3) is partially mitigated by the MHA and Entropy loss. The scores are improved by 3% in Seen and $3-5\%$ in Unseen. Including audio dropout alongside MHA displays even further improvement in the Unseen set by 1%. But the same experiments show a slight drop in the Seen set scores by 1%. It is therefore evident that the model with MHA, Entropy loss and audio dropout is slightly underfitted and can be trained further by tuning the number of epochs and learning rate scheduling.

Dataset	Attention	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Validation Seen	None	60.42	88.34	42.53	67.08	30.65	62.2
	MHA+Entropy	**63.01**	87.56	**45.33**	68.96	**33.03**	63.3
	MHA+Entropy$^\diamond$	62.13	**88.92**	43.58	**69.13**	31.61	**64.87**
Validation Unseen	None	44.27	76.99	21.85	44.02	12.61	36.83
	MHA+Entropy	48.39	81.33	26.01	49.17	15.46	43.36
	MHA+Entropy$^\diamond$	**49.94**	**82.44**	**27.09**	**52**	**16.02**	**45.85**

Table 7.23: All modality attention Experiments ($\diamond$ = with audio dropout of 0.5)

The figures from the test server submissions (Table 7.24) are consistent with our observations on the validation set. The "MHA+Entropy" models are the ones with attended audio. The attention experiments outperform all three baselines of RGB, Optical Flow and audio combinations. Due to the challenge deadline, we were unable to submit our UA and PA models to the test server. Our final submission which is ranked 8-th on the public leaderboard (Figure 7.12) is trained with an additional Action class. Action classes were constructed by considering unique combinations of Verb and Noun classes in the training set. This improves the Action accuracy by approximately 2%. Our scores are achieved while being trained with fewer data compared to the other work with better scores [55]. We also do not use any ensemble unlike the other models [48, 44].

Figure 7.14 displays an instance of inferred attention weights over 3 temporal

segments by an RGB+Audio model, trained with MHA and Entropy loss. The first two attentions focus on the running water, while the last one falls off as the sound of running water stops. The prediction, despite not being highly confident, turns out to be correct in this case. Confusion between the Verbs "close" and "turn-off" is observed among the predictions for all "close tap" annotations in the Unseen validation set (Figure 7.13).

In retrospect, the results prove that selection of a larger audio segment negates the need to sample asynchronously between RGB and audio modalities since most annotations in the Epic-Kitchens dataset are less than 2.1 seconds. The majority of the learned attentions are centrally located proving that for Action Recognition it is enough to fuse audio-visual modalities by considering a relatively synchronous pair of samples. The attention mechanism provides flexibility in such a selection as it will learn to filter out the irrelevant sections if a large audio sample is chosen.

Dataset	Models	Verb		Noun		Action	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Test Seen	R+F	58.63	88.73	41.44	66.57	29.81	48.64
	R+A	55.92	86.33	38.57	64.66	26.54	47.09
	R+F+A	59.61	88.44	41.35	67.19	29.46	50.53
	R+F+A (MHA+Entropy)	62.88	90.28	43.81	**69.52**	32.25	54.04
	R+F+A (MHA+Entropy†)	**63.15**	**90.39**	**43.75**	63.54	32.31	54.39
	R+F+A (MHA+Entropy*)	62.86	89.64	43.44	69.24	**34.53**	**56.72**
Test Unseen	R+F	45.85	76.48	24.55	47.32	15.43	28.58
	R+A	44.11	75.79	23.42	45.54	14.17	28.61
	R+F+A	49.23	77.36	26.25	50.39	17.17	32.16
	R+F+A (MHA+Entropy)	51.31	78.9	**27.72**	52.34	18.81	35.09
	R+F+A (MHA+Entropy†)	51.49	**79.24**	27.59	**52.48**	18.3	34.65
	R+F+A (MHA+Entropy*)	**51.62**	79.14	26.73	52.41	**20.45**	**38.17**

Table 7.24: Test Server Results (R = RGB, F = Flow, A = Audio, † = asynchronous sampling during training, ⋆ = trained with additional Action class)

Seen Kitchens (S1)

#	User	Entries	Date of Last Entry	Team Name	Top-1 Accuracy (%)			Top-5 Accuracy (%)			Precision (%)			Recall (%)		
					Verb	Noun	Action	Verb	Noun	Action	Verb	Noun	Action	Verb	Noun	Action
1	wasun	14	05/28/20	UTS-Baidu	70.41 (1)	52.85 (1)	42.57 (1)	90.78 (4)	76.62 (2)	63.55 (2)	60.44 (4)	47.11 (1)	24.94 (3)	45.82 (4)	50.02 (1)	26.93 (2)
2	action_banks	18	05/29/20	NUS_CVML	66.56 (6)	49.60 (4)	41.59 (2)	90.10 (5)	77.03 (1)	64.11 (1)	59.43 (7)	45.62 (3)	25.37 (1)	41.65 (8)	46.25 (4)	26.98 (1)
3	Sudhakaran	50	05/29/20	FBK_HuPBA	68.68 (3)	49.35 (5)	40.00 (3)	90.97 (3)	72.45 (5)	60.23 (4)	60.63 (3)	45.45 (4)	21.82 (6)	47.19 (2)	45.84 (5)	24.34 (4)
4	tnet	34	05/27/20	SAIC_Cambridge	69.43 (2)	49.71 (3)	40.00 (3)	91.23 (2)	73.18 (3)	60.53 (3)	60.01 (5)	45.74 (2)	24.95 (2)	47.40 (1)	46.78 (3)	25.27 (3)
5	aptx4869lm	12	01/30/20	GT-WISC-MPI	68.51 (4)	49.96 (2)	38.75 (4)	89.33 (8)	72.30 (6)	58.99 (5)	51.04 (16)	44.00 (6)	23.70 (5)	43.70 (7)	47.32 (2)	23.92 (5)
6	weiyaowang	14	05/28/20		66.67 (5)	48.48 (6)	37.12 (5)	88.90 (9)	71.36 (7)	56.21 (8)	51.86 (14)	41.26 (7)	20.97 (7)	44.33 (6)	44.92 (6)	21.48 (8)
7	TBN_Ensemble	1	07/20/19	Bristol-Oxford	66.10 (7)	47.88 (7)	36.66 (6)	91.28 (1)	72.80 (4)	58.62 (6)	60.73 (2)	44.89 (5)	24.01 (4)	46.81 (3)	43.88 (7)	22.92 (6)
8	cvg_uni_bonn	21	05/27/20	CVG Lab Uni Bonn	62.86 (8)	43.44 (10)	34.53 (7)	89.64 (6)	69.24 (8)	56.73 (7)	52.82 (13)	38.81 (11)	19.21 (10)	44.72 (5)	39.50 (10)	21.80 (7)

(a) Seen Kitchens Leaderboard

Unseen Kitchens (S2)

#	User	Entries	Date of Last Entry	Team Name	Top-1 Accuracy (%)			Top-5 Accuracy (%)			Precision (%)			Recall (%)		
					Verb	Noun	Action	Verb	Noun	Action	Verb	Noun	Action	Verb	Noun	Action
1	wasun	14	05/28/20	UTS-Baidu	60.43 (1)	37.28 (2)	27.96 (1)	83.07 (1)	63.67 (2)	46.81 (1)	35.23 (2)	32.60 (1)	17.35 (1)	28.97 (3)	32.78 (2)	19.82 (2)
2	aptx4869lm	12	01/30/20	GT-WISC-MPI	60.05 (2)	38.14 (1)	27.35 (2)	81.97 (2)	63.81 (1)	45.24 (3)	33.59 (6)	31.94 (2)	16.52 (3)	29.30 (2)	33.91 (1)	20.05 (1)
3	action_banks	18	05/29/20	NUS_CVML	54.56 (6)	33.46 (6)	26.97 (3)	80.40 (7)	60.98 (4)	46.43 (2)	33.60 (5)	30.54 (4)	14.99 (5)	25.28 (7)	28.39 (6)	17.97 (4)
4	weiyaowang	14	05/28/20		58.25 (3)	36.74 (3)	26.56 (4)	81.29 (5)	60.33 (5)	43.63 (5)	33.66 (4)	27.09 (6)	15.71 (4)	28.09 (4)	32.34 (4)	18.89 (3)
5	Sudhakaran	50	05/29/20	FBK_HuPBA	56.67 (5)	33.90 (5)	25.71 (5)	81.87 (3)	59.68 (6)	44.42 (4)	30.72 (8)	27.25 (5)	12.74 (6)	25.09 (8)	29.46 (5)	17.93 (5)
6	tnet	34	05/27/20	SAIC_Cambridge	57.60 (4)	34.69 (4)	24.62 (6)	81.84 (4)	61.25 (3)	41.38 (6)	37.81 (1)	30.81 (3)	16.61 (2)	30.37 (1)	32.40 (3)	17.81 (6)
7	TBN_Ensemble	1	07/20/19	Bristol-Oxford	54.46 (7)	30.39 (7)	21.99 (7)	81.22 (6)	55.68 (7)	40.59 (7)	32.56 (7)	21.67 (8)	9.83 (11)	27.60 (5)	25.58 (7)	13.52 (7)
8	cvg_uni_bonn	21	05/27/20	CVG Lab Uni Bonn	51.62 (11)	26.73 (10)	20.45 (8)	79.14 (10)	52.41 (9)	38.17 (8)	30.19 (9)	21.49 (10)	9.77 (12)	27.50 (6)	23.88 (9)	13.50 (8)

(b) Unseen Kitchens Leaderboard

Figure 7.12: Epic Kitchens Action Recognition challenge, CVPR 2020, public leaderboard

index	vid_id	gt_verb	gt_noun	pred_verb	pred_noun	entropy
1	P25_04	close	tap	pour	water	0.46238
2	P25_04	close	tap	pour	water	0.10597
3	P25_04	close	tap	close	tap	0.32649
4	P25_05	close	tap	turn-off	tap	0.14656
5	P25_05	close	tap	adjust	hob	0.01901
6	P25_05	close	tap	pour	water	0.06363
7	P25_05	close	tap	turn-off	tap	0.0064
8	P25_09	close	tap	close	tap	0.3476
9	P25_09	close	tap	turn-off	tap	0.11314
10	P25_10	close	tap	turn-off	tap	0.20476

Figure 7.13: Sample predictions for "Close Tap" from Unseen validation set (gt = ground truth, pred = predicted)

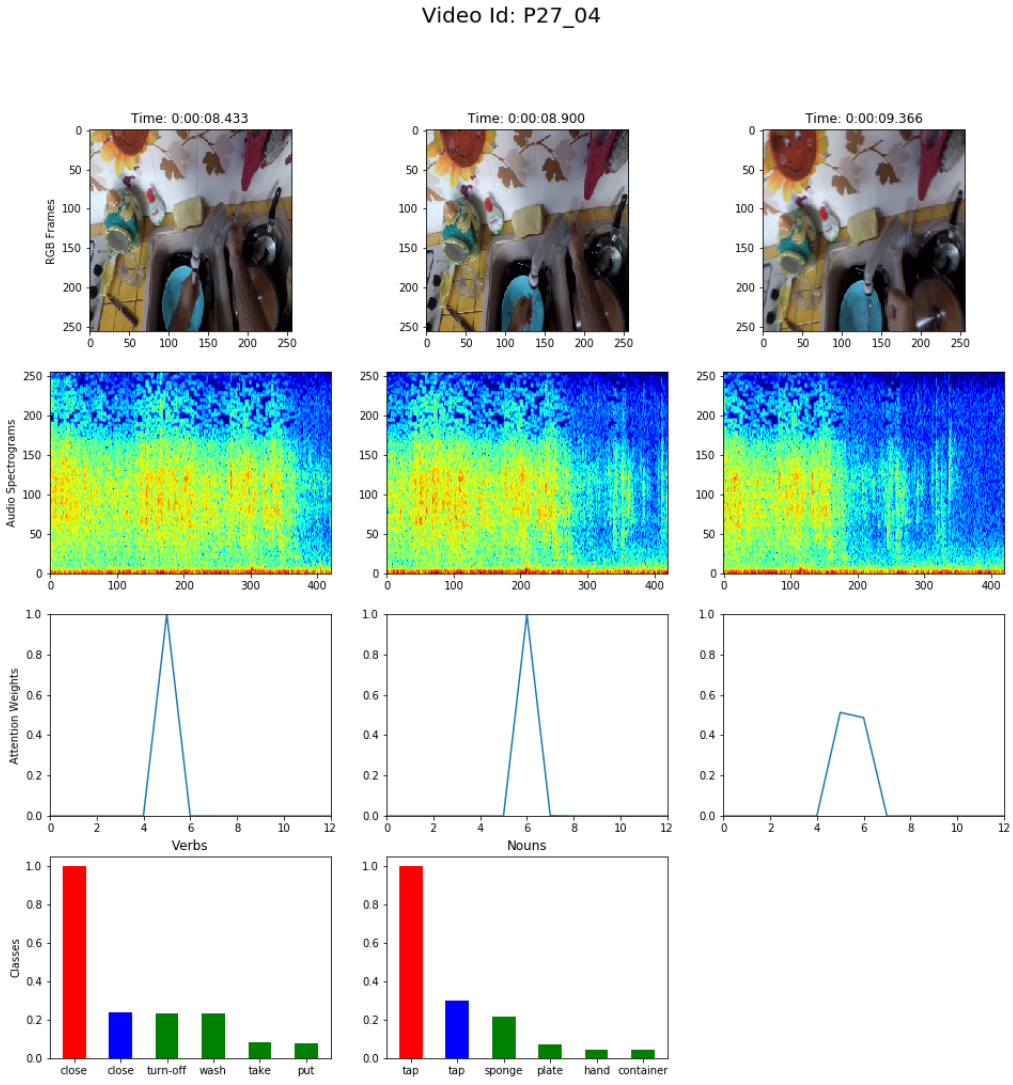


Figure 7.14: Sample from Visualization Module for a "Close Tap" instance

CHAPTER 8

Summary and Future Work

To summarize our work, we see audio can be a useful tool for Action Recognition tasks from videos. It can also provide complementary information to other modalities provided that the different learning speeds are countered to reduce the overfitting. Our attention mechanisms help in this regard apart from inducing flexibility while selecting the temporal window length for audio sampling. The attended audio can be used to replace the asynchronous sampling mechanism within the temporal segments. We observe that most of the attention weights are located centrally *i.e.* synchronous with respect to the visual modality. However, we do not find any asynchronous interactions being learned between the RGB frames and audio. Due to this, we still need to be careful while selecting the audio length, lest it is too large and consumes more memory, without providing any advantage.

In the future, the work on audio can be scaled by either researching on a combination of better audio representations, Deep Learning models created specifically for classification with spectrograms or creating pre-trained audio networks similar to those present for images. The attention mechanism could be improved by looking at early fusion to also localize on specific frequencies instead of only temporal segments.

In conclusion, we successfully prove the effectiveness of audio in Action Recognition from egocentric videos and carry out extensive experiments on this effect with both attended and non-attended audio-visual fusion. We introduce a new framework which allows one to carry out further experiments on top of our work [1]. Our results place us in the top-10 of the Epic-Kitchens, Action Recognition challenge, CVPR 2020, public leaderboard.

[1] `https://github.com/tridivb/attention_based_tbn`

List of Figures

List of Tables

Bibliography

[1] S. S. Stevens, John E. Volkmann, and E. B. Newman. "A Scale for the Measurement of the Psychological Magnitude Pitch". In: 1937.

[2] Jont B. Allen and Lawrence R. Rabiner. "A unified approach to short-time Fourier analysis and synthesis". In: *Proceedings of the IEEE*. Vol. 65. 1977, pp. 1558–1564.

[3] Léon Bottou. "Online Algorithms and Stochastic Approximations". In: 1998.

[4] Christian Schüldt, Ivan Laptev, and Barbara Caputo. "Recognizing human actions: a local SVM approach". In: *International Conference on Pattern Recognition (ICPR)*. 2004, pp. 32–36.

[5] Min Xu et al. "HMM-Based Audio Keyword Generation". In: *Pacific-Rim Conference on Multimedia (PCM)*. 2004.

[6] Christopher M. Bishop. "Pattern Recognition and Machine Learning". In: Springer, 2006, pp. 48–51.

[7] Lena Gorelick et al. "Actions as Space-Time Shapes". In: *Transactions on Pattern Analysis and Machine Intelligence*. Vol. 29. 2007, pp. 2247–2253.

[8] J. Deng et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009.

[9] Honglak Lee et al. "Unsupervised feature learning for audio classification using convolutional deep belief networks". In: *International Conference on Neural Information Processing Systems (NIPS)*. 2009.

[10] H. Kuehne et al. "HMDB: a large video database for human motion recognition". In: *Proceedings of the International Conference on Computer Vision (ICCV)*. 2011.

[11] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild". In: *ArXiv*. Vol. abs/1212.0402. 2012.

[12] Johannes Andreas Stork et al. “Audio-based human activity recognition using Non-Markovian Ensemble Voting”. In: *International Symposium on Robot and Human Interactive Communication (RO-MAN)*. 2012, pp. 509–514.

[13] Lasitha Piyathilaka and Sarath Kodagoda. “Human Activity Recognition for Domestic Robots”. In: *Conference on Field and Service Robotics (FSR)*. 2013.

[14] Karen Simonyan and Andrew Zisserman. “Two-Stream Convolutional Networks for Action Recognition in Videos”. In: *International Conference on Neural Information Processing Systems (NIPS)*. 2014, pp. 568–576.

[15] Max Jaderberg et al. “Spatial Transformer Networks”. In: *International Conference on Neural Information Processing Systems (NIPS)*. 2015, pp. 2017–2025.

[16] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *International Conference on Learning Representations (ICLR)*. 2015.

[17] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[18] Aäron van den Oord et al. “WaveNet: A Generative Model for Raw Audio”. In: *ArXiv*. Vol. abs/1609.03499. 2016.

[19] Christian Szegedy et al. “Rethinking the Inception Architecture for Computer Vision”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2818–2826.

[20] Limin Wang et al. “Temporal Segment Networks: Towards Good Practices for Deep Action Recognition”. In: *European Conference on Computer Vision (ECCV)*. 2016.

[21] Relja Arandjelovic and Andrew Zisserman. “Look, Listen and Learn”. In: *International Conference on Computer Vision (ICCV)*. 2017, pp. 609–617.

[22] João Carreira and Andrew Zisserman. “Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 4724–4733.

[23] Priya Goyal et al. “Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour”. In: *ArXiv*. Vol. abs/1706.02677. 2017.

[24] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. “Learning Spatio-Temporal Features with 3D Residual Networks for Action Recognition”. In: *International Conference on Computer Vision Workshops (ICCVW)*. 2017, pp. 3154–3160.

[25] Shawn Hershey et al. “CNN architectures for large-scale audio classification”. In: *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017, pp. 131–135.

[26] Eric Jang, Shixiang Gu, and Ben Poole. “Categorical Reparameterization with Gumbel-Softmax”. In: *International Conference on Learning Representations (ICLR)*. 2017.

[27] Will Kay et al. “The Kinetics Human Action Video Dataset”. In: *ArXiv*. Vol. abs/1705.06950. 2017.

[28] Justin Salamon and Juan Pablo Bello. “Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification”. In: *IEEE Signal Processing Letters*. Vol. 24. 2017, pp. 279–283.

[29] Ashish Vaswani et al. “Attention is All You Need”. In: *International Conference on Neural Information Processing Systems (NIPS)*. 2017, pp. 6000–6010.

[30] Yunbo Wang et al. “Spatiotemporal Pyramid Network for Video Action Recognition”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2097–2106.

[31] Relja Arandjelovic and Andrew Zisserman. “Objects that Sound”. In: *European Conference on Computer Vision (ECCV)*. 2018.

[32] Dima Damen et al. “Scaling Egocentric Vision: The EPIC-KITCHENS Dataset”. In: *European Conference on Computer Vision (ECCV)*. 2018.

[33] Sander Dieleman, Aäron van den Oord, and Karen Simonyan. “The Challenge of Realistic Music Generation: Modelling Raw Audio at Scale”. In: *International Conference on Neural Information Processing Systems (NIPS)*. 2018, pp. 8000–8010.

[34] Bruno Korbar, Du Tran, and Lorenzo Torresani. “Cooperative Learning of Audio and Video Models from Self-Supervised Synchronization”. In: *International Conference on Neural Information Processing Systems (NIPS)*. 2018.

[35] Yin Li, Miao Liu, and James M. Rehg. “In the Eye of Beholder: Joint Learning of Gaze and Actions in First Person Video”. In: *European Conference on Computer Vision (ECCV)*. 2018.

[36] Andrew Owens and Alexei A. Efros. “Audio-Visual Scene Analysis with Self-Supervised Multisensory Features”. In: *European Conference on Computer Vision (ECCV)*. 2018.

[37] Naoya Takahashi, Michael Gygli, and Luc J. Van Gool. “AENet: Learning Deep Audio Features for Video Analysis”. In: *IEEE Transactions on Multimedia*. Vol. 20. 2018, pp. 513–524.

[38] Yapeng Tian et al. “Audio-Visual Event Localization in Unconstrained Videos”. In: *European Conference on Computer Vision (ECCV)*. 2018.

[39] Alejandro Cartas et al. "How Much Does Audio Matter to Recognize Egocentric Object Interactions?" In: *ArXiv*. Vol. abs/1906.00634. 2019.

[40] Alejandro Cartas et al. "Seeing and Hearing Egocentric Actions: How Much Can We Learn?" In: *International Conference on Computer Vision Workshop (ICCVW)*. 2019, pp. 4470–4480.

[41] Jiaxu Chen et al. "An End-to-End Audio Classification System based on Raw Waveforms and Mix-Training Strategy". In: *Conference of the International Speech Communication Association (INTERSPEECH)*. 2019.

[42] Ali Diba et al. "Holistic Large Scale Video Understanding". In: *ArXiv*. Vol. abs/1904.11451. 2019.

[43] Christoph Feichtenhofer et al. "SlowFast Networks for Video Recognition". In: *International Conference on Computer Vision (ICCV)*. 2019, pp. 6201–6210.

[44] Evangelos Kazakos et al. "EPIC-Fusion: Audio-Visual Temporal Binding for Egocentric Action Recognition". In: *International Conference on Computer Vision (ICCV)*. 2019.

[45] Yuan Liu et al. "AI for Earth: Rainforest Conservation by Acoustic Surveillance". In: *ArXiv*. Vol. abs/1908.07517. 2019.

[46] Lili Meng et al. "Interpretable Spatio-Temporal Attention for Video Action Recognition". In: *International Conference on Computer Vision Workshop (ICCVW)*. 2019, pp. 1513–1522.

[47] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *International Conference on Neural Information Processing Systems (NIPS)*. 2019, pp. 8026–8037.

[48] Swathikiran Sudhakaran, Sergio Escalera, and Oswald Lanz. "Hierarchical Feature Aggregation Networks for Video Action Recognition". In: *ArXiv*. Vol. abs/1905.12462. 2019.

[49] Swathikiran Sudhakaran, Sergio Escalera, and Oswald Lanz. "LSTA: Long Short-Term Attention for Egocentric Action Recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 9946–9955.

[50] Du Tran et al. "Video Classification With Channel-Separated Convolutional Networks". In: *International Conference on Computer Vision (ICCV)*. 2019, pp. 5551–5560.

[51] Ruohan Gao et al. "Listen to Look: Action Recognition by Previewing Audio". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[52] Sanjeel Parekh et al. "Weakly Supervised Representation Learning for Audio-Visual Scene Analysis". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing*. Vol. 28. 2020, pp. 416–428.

[53] Helena Peic Tukuljac et al. *SpectroBank: A filter-bank convolutional layer for CNN-based audio applications*. 2020. URL: `https://openreview.net/forum?id=HyewT1BKvr`.

[54] Weiyao Wang, Du Tran, and Matt Feiszli. "What Makes Training Multi-Modal Classification Networks Hard?" In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[55] Xiaohan Wang et al. "Symbiotic Attention with Privileged Information for Egocentric Action Recognition". In: *Association for the Advancement of Artificial Intelligence (AAAI)*. 2020.

[56] Fanyi Xiao et al. "Audiovisual SlowFast Networks for Video Recognition". In: *ArXiv*. Vol. abs/2001.08740. 2020.

Abbreviations

AR Action Recognition. vii, 1, 2, 10, 18, 24, 25, 30, 47, 52, 55, 58

ATBN Attention-based Temporal Binding Network. 27, 29, 39, 57

CE Cross-Entropy. 24, 25

CNN Convolutional Neural Network. 3, 4, 5, 6, 10, 12, 17, 27, 30, 33

CVPR Conference on Computer Vision and Pattern Recognition. 52, 55, 58

FC Fully Connected. 3, 4, 6, 17, 18, 21, 22, 31, 57

MHA Multi-headed Attention. 12, 18, 20, 21, 41, 42, 43, 45, 46, 47, 48, 49, 50, 51, 57, 58, 59

NLP Natural Language Processing. 12

PA Prototype Attention. 18, 22, 23, 24, 31, 48, 49, 50, 57, 58, 59

PE Positional Encoding. 21, 22, 31, 40, 41, 42, 46, 47, 57, 59

ReLU Rectified Linear Unit. 5, 18, 57

SGD Stochastic Gradient Descent. 6, 26, 31

STFT Short-time Fourier transform. 7, 30

TBN Temporal Binding Network. vii, 2, 10, 11, 14, 27, 35, 57

TBW Temporal Binding Window. 27

TSN Temporal Segment Network. 10, 11, 14, 57

UA Unimodal Attention. 18, 21, 22, 23, 46, 47, 48, 49, 50, 57, 58, 59

VA Video Analytics. 9