Audiovisual Active Speaker Detection: G056 report

G056 (s1707651, s1661399, s1645982)

Abstract

Active speaker detection is an integral element in many applications dealing with spoken information. The team at Google AI Perception have recently presented the comprehensive audio-video AVA-ActiveSpeaker dataset [1], facilitating robust model building and enabling future comparison. They also set the benchmark performance by presenting a novel approach, additionally capable of real-time detection. The aim of this project is to take forth the work of [1] and investigate potential strategies of optimising audiovisual active speaker detection, whilst conscious of the limited computational resources imposed by real-time processing.

The key contributions of the project include a comprehensive empirical evaluation of audio and visual embedding networks. The results have shown that the proposed `v3_slim` model has made a 20% improvement reaching 0.70 mAP over the baseline derived from [1]. Moreover, building on `v3_slim`, `v3_sa` employing asymmetric audio and visual embeddings made an additional significant increase in mAP reaching 0.75. Further improving on `v3_sa`, `v3_sync` using embeddings pretrained with contrastive loss reached 0.86 mAP. It was concluded that employing a MobileNet-V3 [2] architecture and using pretrained asymmetric embeddings results in substantial performance improvements.

1. Introduction

The ability to detect the currently active speaker(s) is a stepping stone to a truly diverse range of applications. Speaker diarisation can be used for conversational indexing, which can then be expanded to various forms of information retrieval, such as speech-to-text transcription, translation, or even movie narrative comprehension [3; 4]. This ability of unsupervised labelling has also been used in autonomous dataset mining [5; 6; 7]. Similarly to how the human brain acts in noisy environments, it is possible to identify and focus on a particular speaker, enhancing the perception of their voice [8; 9]. Spatial awareness is of great importance to high-level human-robot interaction, as more natural and human-like behaviour can be achieved [10; 11]. Further speaker identification can enable richer conversation as responses can be personalised. Likewise, spatial awareness can be employed during conferential events for automatic camera direction [12], or targeted subtitling [13].

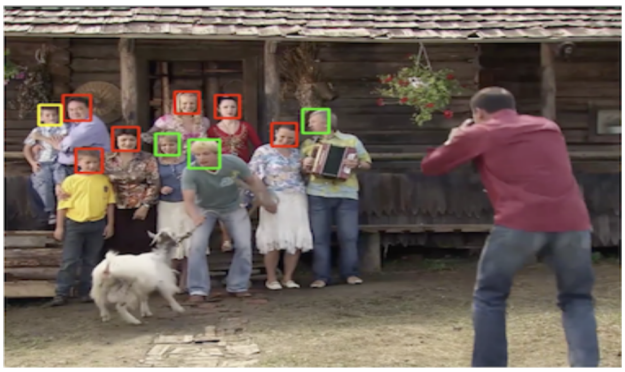

Figure 1. Visualizations of overlapping speaker instances. A green bounding box represents speaking and audible, yellow represents speaking and inaudible, red represents not speaking [1].

Previous work typically differs in the underlying data source of the algorithm. Audio-only data is common among speaker diarisation implementations, as clustering techniques, where each cluster corresponds to an individual speaker, have been shown to be suffice in controlled scenarios [14]. However, as observed in a speaker diarisation research review done by Anguera et al., [14], such methods experience severe variation in performance given different test data. It was also noted that overlapping speaking episodes, such as in Figure 1, pose great challenges in the domain.

Indeed, using audio signal alone can be rather limiting, prompting expansion to other sources of information. For pre-recorded scenarios, it is possible to additionally use transcripts [3; 4], however a more viable strategy is the utilisation of video signal. Visual-based implementations achieve satisfactory results in their own right, and have been shown to prevail during episodes of overlapping speakers [10; 11; 1]. Nonetheless, they too possess vulnerable scenarios, where certain facial movement can be mistaken for speaking [1].

As such, fusion of audio and video signal has also been thoroughly explored. A statistical approach for speaker localisation based on correlation between audio and regions of video signals was proposed in [9]. Zhang et al., [15] were able to extract the most relevant features from the audiovisual input via AdaBoost and Decision Stumps to achieve high accuracies in meeting room environments. Stefanov et al., [11] used Hidden Markov Models based on audio and facial movement.

More recently deep learning has been employed in the field, namely Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). The networks are built for each modality, after which their results are either concatenated or the objective function modified for the overall model to consider both mediums. Chung et al., [16] developed an audio-video synchronisation method based on CNNs, where upon fusion of the modalities a contrastive loss function, presented in [17], is used. Essentially, the loss function looks at the Euclidean distance of the audio and video embedding outputs. The learnt data embeddings were shown to be beneficial for other tasks, such as active speaker detection and lip reading. Owens et al., [18] take on a slightly different approach by concatenating the outputs of the individual CNNs that are then fed in to a third CNN. In [19], a single Long Short-Term Memory (LSTM) network, where weights were shared among modalities, was used for speaker detection and identification and even outperformed the CNN baselines.

Unfortunately, *all* aforementioned implementations are evaluated on restricted datasets, that are either hand-crafted and hence small in size, biased towards specific environments, or the combination of the two. For example, in [16] only 6 speakers were evaluated, in [11] the overall dataset was less than an hour long, in [19] and [3] clips from only 1 and 4 TV series were used, respectively. The VoxCeleb dataset produced in [6], on the other hand, is large and contains 153,561 speaking episodes of 1,251 different speakers. However, during unsupervised labeling, conservative detection thresholds were set to minimise the number of false positives. Additionally, frames where no face was detected, e.g., during narration or when people were not facing the camera, were discarded. The resulting high precision, yet low recall, filtering is bound to produce dataset of limited variance. In [7] an even larger dataset was produced, however with an even more restrictive filtering process, additionally discarding non-English video sources and facial frames with closed lips. In [5] a staggering 150,000 distinct speakers were annotated, yet without any background noise, nor overlapping speech episodes. As of result, the absence of a sufficiently large and diverse dataset is generally acknowledged as the main cause of stagnation in terms of generalisation in this domain [1; 14; 10; 5; 7; 6].

The team at Google AI Perception [1] have recently presented the AVA-ActiveSpeaker dataset – a densely hand-labeled dataset that addresses all these issues. Although comparatively not as large, with 38,500 speech episodes, the dataset aims at capturing the most amount of variety possible. It is composed of densely annotated 15-minute clips from 160 movies from around the world. Hence, language, actor demographics. quality and recording conditions are all diverse. During labeling, automatic face tagging was employed, however, once done, it was additionally examined by human annotators to correct occasional mistakes. As of result, the labeling process was not constrained by an underlying algorithm and the most severe filtering was imposed only during selection of said movies, as black and white, cartoon, low resolution and mature videos were discarded [20]. In the same paper [1] a real-time end-to-end audiovisual model, consisting of individual CNN embeddings and a fusing GRU, was presented as a benchmark of performance for the dataset. Authors recognised their lack of exploration in terms of optimal model structure and further announced the "Active Speaker Detection" challenge for the dataset as part of the 4^{th} ActivityNet challenge at CVPR 2019. Only 5 teams participated, however, with the best performance achieved by Chung et al., [21] with a mean average precision (mAP) of 0.878. In the challenge, and consequentially in this project, mAP solely referred to the precision of the `SPEAKING_AUDIBLE` label [22].

The goal of this project is to belatedly and artificially take part in the challenge and investigate different approaches to improving the current solutions in terms of embeddings specifically. Section 2 describes dataset acquisition and handling. Section 3 provides an overview of the various approaches to active speaker recognition. Section 4 delineates and justifies the chosen techniques for potential improvement, while Section 5 presents specific undertaken experiments and their results. Finally, Section 6 concludes the findings of the research.

2. Dataset

The dataset was constructed through a combination of automated face track detection and human annotation of speech. Each video has between 5382 and 44951 annotations for a total of $\sim 3.4M$ annotations. Each annotation defines a face bounding box and speech label for a uniquely identified speaker, at a particular time. The labels include `SPEAKING_AUDIBLE`, `SPEAKING_NOT_AUDIBLE` and `NOT_SPEAKING`.

2.1. Extraction process

The audio streams were extracted from each video using `ffmpeg`. Each stream was extracted as a single channel MP3 at a bitrate of 160KBps and a sampling rate of 44.1KHz. The bitrate was chosen as YouTube streams audio at up to 128KBps in the AAC format, approximately equivalent to 160KBps MP3 [23].

[24] noted that MP3 bitrates higher than 32KBps have minimal effect on speech recognition performance on voice only audio (i.e. without the background noise or overlapping speech present in the AVA dataset). In order to preserve all available audio information at this point, prior to preprocessing, a higher bitrate was used.

2.2. Relation to AVA active speaker challenge

The challenge conducted in 2019 provided a testing server to which model predictions for the test set were submitted. The true labels for the test set were not available, meaning that training, validation, and test sets were formed from the original training data.

Whilst the CVDFoundation hosted all 160 videos [25], there

were only annotation files for 153 videos. These contained $\sim 3.4M \simeq 93\%$ of the original $3.65M$ labels. Due to the limited resources and time available, a subset of the 153 available videos was used. The frame rates of the videos varied between 23 and 30 frames per second (fps) whilst the annotations were all listed at 25fps. Rather than duplicating or removing frames to match the annotation sets, 100 videos were chosen from the 107 videos which were encoded at 25fps. A random 65/15/20 train/validation/test split was then constructed.

2.3. Frame resolutions

The resolutions of the extracted face frames vary broadly. Figure 2 shows the heavily skewed distribution of resolutions. The 128×128 resolution used by the Google AI Perception team in [1] was determined to be a reasonable resolution, maintaining necessary information whilst being small enough for performant computation with limited resources. The images were resized using the interpolation methods recommended by OpenCV [26] for upscaling and downscaling images.

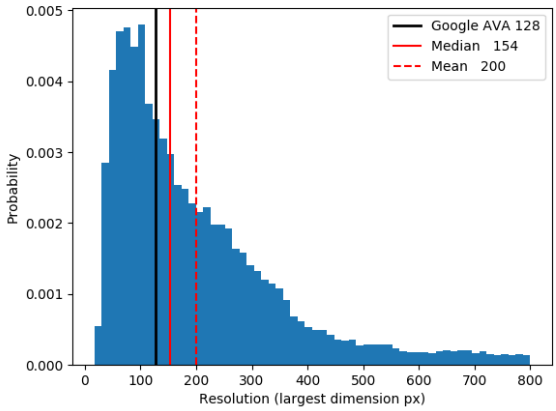

Figure 2. Distribution of face frame resolutions (Upper $\sim$ 2% trimmed for clarity)

A further issue to be considered is that the face frames are a mixture of profile and front on images. This range of angles requires that the model learns how to extract a range of equivalent features which appear differently, i.e an open mouth looks different in profile but corresponds to the same feature.

2.4. Performant loading of a large dataset

Due to the RAM limits and lack of high speed storage on the MLP cluster, Google Compute Engine was used. The extracted frames were ill-suited to loading from disk, as the selection of frames for a batch results in many small random reads. This was a clear bottleneck when training began, as the majority of the training time was data loading whilst the GPUs remained idle.

The images were first written in groups to LMDB[27], a memory mapped database allowing zero-copy reads, allowing saturation of the 240MB/s read speed of Google Cloud Compute datacentre local SSDs. A pair of local PCI-E SSDs sped up training but were saturated at 1.4GB/s but were still a bottleneck. Attaching local SSDs also limited the availability of GPUs, meaning that suitable machines were not always available

Finally, the local SSDs were replaced with a high memory machine. Python 3.8's shared memory was used to load the 105GB dataset into memory in a separate process, allowing the subsequent training processes to access them without any disk IO. This allowed for full GPU utilisation during training.

3. Related Work

This section outlines and discusses literatures relevant to active speaker detection. Different data pre-processing techniques used for speaker recognition are delineated. Cross-modal approaches, which have been common to enable audio-visual correspondence of speech [28], are also explored. Embedding networks are further reviewed in the context of cross-modal classifiers to encode underlying information contained in the video frames and the audio signals. Moreover, some multimodal decoding predictor networks used in literatures are examined.

3.1. Pre-processing

Pre-processing, like feature extraction and different filtering techniques, has been deemed as a crucial step for active speaker detection since for signal processing tasks it can have a considerable impact on the performance of systems [29]. Features that carry particular information on speakers and data that has reduced noise through filtering can aid speaker discrimination. Earlier surveys [29; 14] have shown that studies have placed special attention on feature selection as signals tend to have irrelevant components for speaker recognition. Audio features ubiquitous in speech, invariant against morphemes but variant across speakers and that are easy to extract from raw signals have all been employed for speaker discrimination [30]. On the other hand, while still essential for enhanced performance, with the rise of deep learning detailed manual feature extraction has become more peripheral as CNNs have become better at such tasks [6].

In [31] it is suggested that feature extraction techniques are necessary for audio inputs since one-dimensional raw signals are too erratic and Fourier transforms are too complex for DNNs to approximate. MFCCs capture short-term spectral information using the Mel Frequency Scale, which aims to represent sound waves as the human auditory system would perceive them. Thus, MFCCs allow the retention of information that characterise phonetic speech while discarding other information [31]. However, as pointed out in [6; 29; 14], MFCCs are not always an optimal choice as they may not be able to convey long-term speaker information well, and their performance may also be impeded by noisy signals.

Long-term speech features, such as prosodic features capturing auditory qualities of sound, in combination with MFCCs may be able to help leverage the inadequacies of MFCCs alone in the context of speaker discriminability [32; 29]. Imseng and Friedland outlined several long-term features that have good speaker discriminability including pitch, formants and harmonics [33].

A more recent study [34] focused on the domain of speaker verification, proposing a new method for integrating prosodic features and mitigating the shortcomings of MFCCs. Mel-frequency spectral coefficients (MFSCs) were used, which preserve locality of the frequency domain signal as opposed to MFCCs. Furthermore, they constructed a network architecture where a CNN network, receiving the MFSCs as input, is accompanied by an MLP network, receiving 18 prosodic features as input, which is then combined at a joint representation layer.

As long-term features may be more difficult to extract resulting in a delay in decision making, the choice of features largely depends on the desired attributes of the model. As stated in [30], a trade-off is to be made to balance between speaker discriminability, robustness and pragmatism.

Other audio pre-processing techniques for aiding speaker recognition include normalisation techniques. The study assembling the VoxCeleb dataset applied mean and variance normalisation on the frequency bins of the spectrum. Nagrani et al. claim that such normalisation methods allowed for 10% growth in classification accuracy [6].

Visual pre-processing steps have also been applied to input face thumbnails in [7]. Gaussian filtering was used for face landmark smoothing, which was claimed to be essential for increased performance.

3.2. Embeddings and Predictor Networks

There have been a wide range of embedding networks proposed in literature to extract essential information from videos. The majority of such embedding networks are CNNs and are symmetrical across modalities [35; 21; 36].

Joon et al. proposed SyncNet for audio-video synchronisation in [16], which applied audio and video embedding networks for the synchronisation by matching speech to speaker lip movement. It was suggested that the model could be applied to active speaker detection, where for a speaker to be active a given speech-lip sync threshold is to be met. SyncNet uses two CNN embeddings networks, however, instead of a predictor network following the embedding networks, contrastive loss applied, which uses the Euclidean distance between the embeddings [16]. SyncNet was further improved in [35], where the aforementioned pairwise loss is replaced by multi-way matching, which incorporates sequential information in addition to the distance between embedding pairs. This new loss function receives one input feature from the visual embedding and multiple features from the audio embedding. This has allowed for substantial improvement in lip synchronisation, however, on active speaker recognition only a marginal improvement of 0.1% was seen.

Joon Son Chung, the lead author of the lip synchronisation related papers, also participated in the ActivityNet Challenge 2019, where he attained the best mean average precision for active speaker detection. He employed pretrained embeddings borrowed from [35], however instead of applying Euclidean distance on the two embeddings, a backend predictor network was employed. Two types of networks were experimented with: one consisting of two bidirectional LTSMs and the other consisting of two temporal convolutions. The output of these are then concatenated and fed into a fully connected classifier layer [21].

Another research that was concerned with cross-modal learning is [36]. One of its main objectives was embedding audio and visuals into a shared space that would enable cross-modal retrieval. It further aimed at locating the source of a sound within an image. The research proposed the AVE-Net which again consists of two parallel embedding networks and a subsequent predictor network. AVE-Net takes a single input video frame and one second of corresponding audio in the form of a log-spectrogram. The embedding networks consist of an Audio and an Image ConvNet for the audio and visual embeddings respectively, followed by a pooling layer two fully connected layers and L2 normalisation. The AVE-Net was used as a base of comparison in [35]. In both researches audio-visual correspondence is relied upon, thus the predictor network computes the Euclidean distance between the two embeddings [36], however, AVE-Net includes an extra fully connected layer before the softmax and cross entropy loss. Moreover, [35] uses multi-way cross entropy loss, whereas AVE-Net employs two-way cross entropy loss.

Most cross-modal embedding networks have symmetric structures. Choosing more tailored networks for each of the two modalities has been experimented with in the current paper. The potential application of more specialised state-of-the-art embedding networks, such as FaceNet [37] or other more recent ResNet based facial recognition networks outlined in [38] for the encoding of speakers' faces is yet to be explored.

Predictor networks following the embedding networks have also been used in several studies [36; 1; 21]. These networks usually have simpler structures than the embedding ones. A simple fully connected layer preceded by normalised Euclidean distance between the embeddings in the case of the AVE-Net is reasonable given that good quality embeddings map inputs into the same feature space [36]. Other types of more complex RNNs were utilised in some studies, such as GRUs in [1], or bi-directional LSTMs in [21]. RNNs have deemed to be advantageous given the sequential nature of the AVA dataset, however, experiments with such predictor networks were outside the scope of the current study.

4. Methodology

Bearing in mind that embeddings contain the majority of parameters in multi-modal models, it was decided to focus on embeddings specifically. This section will introduce the baseline given in the dataset's parent paper [1], and expand on the strategies of potential improvement of the embedding performance.

4.1. Baseline

The audio and visual embedding networks are based on MobileNet-V1 proposed in [39]. Instead of conventional convolutions, MobileNet-V1 utilises depthwise separable convolutions (DwSs), capable of near state of the art performance at significantly decreased computational demand. While standard convolution extracts and combines features simultaneously, DwS factorises the process in to two steps, namely depthwise and pointwise convolution. The former applies a single filter per input channel, whereas the latter computes linear combinations of the depthwise convolution through simple 1x1 convolution, generating the desired number of output channels. As a result, the multiplicative interaction between filter size (D_K) and number of output channels (N) is broken, leading to DwS blocks being $D_K^2 + N$ times more efficient than their standard counterpart [39]. Such lightweight architecture enables compact yet powerful model-building, making real-time active speaker recognition possible.

The embedding networks sequentially receive input representing 200-millisecond segments of video. In [1] audio is represented via Mel-spectograms, however in this work MFCCs are used instead, as they are more compact and are a popular choice for speaker detection and diarisation [29]. Stereo audio segments are merged in to one channel and 19×13 MFCCs are computed. Visual input is represented by 5 consecutive 128×128 grayscale face thumbnails. The output of networks is fused together by concatenation and fed into a predictor network of two fully-connected layers followed by softmax. The paper also presented another more accurate model with a GRU predictor, however considering this project's focus on embeddings, the simple linear layer variant of predictors was used throughout research.

4.2. Improved MobileNet

Since the real-time aspect of speaker detection was enticing to the team and MobileNets have recently experienced improvement, newer iterations of the models were explored.

4.2.1. MobileNet-V2

MobileNet-V2s were presented in [40] with the defining element being the inverted residual block (InvRes). Authors argued that information commonly resides in manifolds of lower dimension and ReLUs are capable of preserving said information, but only if it is in a low-dimensional subspace of the function's input. This lead to the implementation of thin residually-connected bottleneck blocks, inside which the number of channels would be expanded through 1×1 convolution before entering a DwS block that would ultimately contract the number of channels back to the original volume, allowing for residual connections to be made. The expansion acted as a source for the network's expressiveness and the residual connections were used for better gradient flow. It was found that residual connections between the bottlenecks performed significantly better than connections between expansions, further highlighting the importance of low-dimensional encoding. Furthermore, it was noted that when the manifold of interest remains non-zero after ReLU transformation, this corresponds to a linear transformation; as well as it was shown that non-linearity destroys information in low-dimensional space. This lead to the omission of activation functions between the inverted residual blocks. When they were used, it was ReLU6, which is defined by:

$$\text{ReLU6}(x) = \min\left(\text{ReLU}(x), 6\right) \tag{1}$$

The function is known to be more robust with low-precision computation [40]. Overall, this lead to state of the art real-time performance with datasets such as ImageNet and COCO. Although being more computationally demanding than V1 due to the extra 1×1 convolution, the notion of expansion allowed for utilisation of smaller input and output dimensions and hence more efficient memory consumption. On ImageNet, a 3.74 M parameter model achieved 60.3% top-1 accuracy with 17.6 ms latency on a Pixel 1 [41].

4.2.2. MobileNet-V3

In November 2019 MobileNet-V3 was presented [2], displaying further accuracy improvement with even more compact architecture. Among the developments, a variant of the swish activation function, dubbed h-swish, was used. Swish, discovered in [42] and defined in (2), was shown to be consistently superior over ReLU. Nonetheless, due to the sigmoid function, it is not of best fit for mobile usage, and hence h-swish (3) was put forth for V3. Without any difference in accuracy, h-swish is both more precise and optimised.

$$\text{swish}(x) = x \cdot \sigma(x) \tag{2}$$

$$\text{h-swish}(x) = x \cdot \frac{\text{ReLU6}(x+3)}{6} \tag{3}$$

Another addition was the Squeeze and Excite (SE) layer after channel expansion in InvRes. Through convolution, channel interdependencies are entangled with spatial features captured by the filters. Hu et al., [43] suggested that explicit modelling of channel interdependencies can be of great use in convolution. Channel-wise statistics are firstly gathered through global average pooling (squeeze), and are then flattened and fed in to a series of linear layers (excite). The layers consist of a $C \times \frac{C}{r}$ layer, ReLU, $\frac{C}{r} \times C$ layer and a sigmoid for rescaling, where C is the number of channels and r is a parameter of computational cost. Each channel is

then multiplied by their resulting scalar to recalibrate their response to convolutional filters. In V3, fixing r to 4 was found to be most effective.

The last major improvement stemmed from highly optimised structure. Firstly, exhaustive grid search based on a state of the art network generation algorithm, NetAdapt [44], was employed, where the target metric was a balance between model accuracy and latency recorded on real devices. Besides finding pareto-optimal number of layers and their input and output sizes, the search suggested usage of h-swish only in later layers due to decreasing input resolution, and the usage of not only 3×3, but also 5×5 kernels. Secondly, further manual optimisations were made that were not in the scope of the search. These included halving channel-depth of the first layer and adding h-swish as a compromise, as well as performing global average pooling before the last and biggest channel expansion layer[1]. On ImageNet, a 2.9 M parameter model, which is 22% less than the V2 example, achieved 67.5% top-1 accuracy with 15.8 ms latency on a Pixel 1 [41].

The scales of the proposed architectures for both V2 and V3 are too large for the ActiveSpeaker dataset, and so custom architectures will have to be empirically identified. Although the generated structure of V3 cannot be used, the developed suggestions, such as mixed kernel sizes and conservative h-swish usage, will still be exploited.

4.3. Non-symmetric Embeddings

Symmetrical embeddings are commonly used [1; 16; 21; 35]. However, there is no need in such symmetry, especially considering that the image input is of significantly larger size. Too fine-grained decomposition of the audio signal through convolution is likely to lead overfitting, ultimately being counter-productive to fusing modalities. After determining the best performing MobileNet version, it was decided to experiment with downsized audio embeddings.

4.4. SyncNet

Benefits of pre-trained embeddings are obvious – faster training, that can also lead to improved generalisation, as well as tolerance of smaller datasets. However, pre-training can also be done with a specific goal in mind. For example, Chung et al., [16] found that multi-modal synchronisation models, i.e., SyncNets, trained with contrastive loss can be expanded to active speaker detection, as a high synchronisation score[2] between an audio segment and a face-track implies an active speaker. As the metric of focus for the challenge is precision of `SPEAKING_AUDIBLE`, pre-training embeddings with contrastive loss can aid performance. The top predictor will be omitted and the output of the individual embeddings will be fed in to (4), where v and a are video and audio embedding outputs, respectively, y is 1 when the sample is `SPEAKING_AUDIBLE` and 0 otherwise, and *margin* is a threshold for similarity commonly set to 1.

$$E = \frac{1}{N}\sum_{n=1}^{N}(y_n)\cdot d_n^2 + (1-y_n)\cdot \max(margin - d_n, 0)^2 \qquad (4)$$
$$d_n = \|v_n - a_n\|_2$$

Once pre-trained, the embeddings will be used in a complete model with the top predictor and an augmented loss function (5), where L is cross-entropy loss of the output class, E is contrastive loss of the embedding outputs and α and β are scaling factors defaulted at 1.

$$\text{Loss} = \alpha L + \beta E \qquad (5)$$

The addition of E in the loss function will further provoke the model to focus on `SPEAKING_AUDIBLE`.

5. Experiments

This section describes all conducted experimental set-ups and the corresponding results. In Table 1 both validation and test set results are shown, however, all decisions are made based on the validation set. The baseline model will be referred to as `v1`. For V2 and V3 two architectures of the same depth yet different width (i.e., number of channels) will be attempted each. The naming convention for those will be `<version>_<width>`, e.g., `v2_slim`. The network blocks are shown in Tables 2 and the exact architectures are shown in Table 3. After finding the best structure, a reduced audio embedding model, `<version>_sa`, will be attempted. Finally, the best performing model at the current stage will be used as a template for the SyncNet pre-training. The pre-trained embeddings will then be used in the full model, `<version>_sync`, with the loss function defined in 5.

A batch size of 128 samples was chosen, which deemed to be both memory efficient and enabled fast training with relatively minor fluctuations of loss. Additionally, training was conducted for a maximum of 40 epochs. From experience with MobileNet-V1, it was observed that within 40 epochs the model started overfitting, thus early stopping was enabled. The minimum number of epochs to be run was set to 20 and 10 for the models with slimmer and wider embeddings respectively. AdamW optimiser was used across all experiments due to its improved generalisation performance over Adam. The improvement over Adam is made by adding weight decay at a later point such that it does not interfere with the moving averages of the gradient [45].

5.1. Baseline – MobileNet-V1

Symmetrical embedding networks were used, each of which have a regular 2-dimensional 3x3 convolutions outputting 32 channels in the input layer, followed by batch normalisation (BN) and ReLU6 activation function (ConvReLU6 in Table 2). This was followed by seven depthwise separable convolution blocks (DwS), the second of which doubles the

[1]The last large channel expansion has been deemed crucial for rich feature extraction [2].

[2]Or low contrastive loss result.

number of channels to 64. Adaptive 2-dimensional average pooling was performed at the last layer of the embedding networks. However, no pooling was done before the pool, instead, all but the first DwS block had a stride of 2, reducing the size of the feature maps flowing through the network layers.

The baseline model was able to gradually learn in the first 9 epochs, after which it started overfitting as shown in Figure 3. It reached an mAP score of 0.60 and 0.58 on the validation and the test sets respectively. This is about 14% less than what was obtained in [1] – in the paper providing guidance for the establishment of this baseline. However, this result is satisfactory as a baseline given that only a subset of the AVA dataset was used.

5.2. MobileNet-V2

Both models have similar architectures which consist of a ConvReLU6 block, seven inverted residual InvRes blocks and finally two 1×1 ConvReLU6 blocks with an adaptive average pool between them. The ratio between an InvRes block's bottleneck input size and its inner size is known as the expansion ratio (t). The expansion ratio is consistently higher in `v2_wide`, setting the two models apart: in `v2_wide` it goes up to $t = 4$, while in `v2_slim` it is capped at $t = 2$. Such architectures were chosen to test the effect of higher-dimensional activation space on generalisation [40].

Both models made clear improvements over the baseline: `v2_wide` and `v2_slim` have reached an mAP of 0.63/0.64 and 0.69/0.69 on the validation/test set respectively. Model `v2_slim` outperformed `v2_wide`, which is due to the wider network overfitting. As observable on Figure 3, just after 7 epochs `v2_wide` started overfitting, while `v2_slim` was able to learn more steadily up until the 14^{th} epoch. This is likely because the wider network retained the smaller intricacies of the data, thus not being able to generalise as well. Therefore, it can be deduced that wider networks may not be as suitable for the current problem given the MobileNet-V2 embedding structure. Furthermore, `v2_slim` took 11% less time on evaluation of the test set than `v1` whilst having 17% more parameters, exemplifying the efficiency of InvRes.

5.3. MobileNet-V3

Both `v3_wide` and `v3_slim` are composed of a ConvReLU6 block, DwS block, six inverted residual blocks of varying expansion ratio and kernel size, a 1×1 ConvReLU6 block with h_swish activated, an AdaptiveAvgPool2d, and finally two further 1×1 (ConvReLU6s) with h_swish.

The models scored similar accuracy and loss to their V2 counterparts, with `v3_slim` making the most notable improvement. `v3_slim` reached a validation/test mAP of 0.71/0.70, higher than `v3_wide`, 0.65/0.65, and the previously highest `v2_slim`, 0.69/0.69. It was again clear that the larger model had overfitted. For the test set, `v3_slim` now took 13% more time, however this can be expected given the additional 8,000 parameters from 5×5 convolutions and SE layers.

5.4. Non-symmetric Embeddings

The `v3_sa` model is based on `v3_slim`, on the clear evidence that it provides the best performance. Each MFCC is of dimension 19×13, and by the final 3 layers this was reduced to 2×1 and then 1×1 whilst the kernel size was 5×5. Therefore, `v3_sa` had 3 less layers and no 5×5 kernels. The change to the audio subnetwork produced a clear improvement across all measures, most notably and importantly in the mAP of 0.76/0.75. A significant improvement was also observed for runtime, as it improved by 28% whilst having 154 less parameters than `v1`.

5.5. SyncNet

With the same justification as before, `v3_sa` was used as the base architecture. The synchronised model achieved mAPs of 0.87/0.86 for validation and test data respectively, a significant increase over `v3_sa` which was itself clearly more performant than previous models. The result highlights the practicality of synchronising embeddings between subnetworks when working with separate but correlated data.

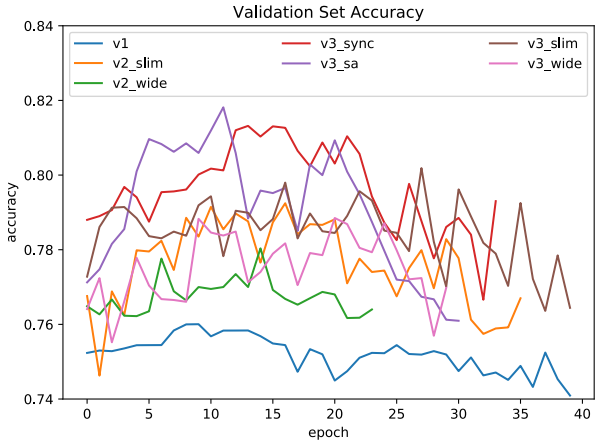

Figure 3. Validation set accuracies per epoch. Early stopping enabled.

Measure	Loss	Accuracy	mAP
Model/Dataset	**Val./Test**	**Val./Test**	**Val./Test**
v1	0.50/0.52	0.76/0.75	0.60/0.58
v2_wide	0.47/0.48	0.78/0.77	0.63/0.64
v2_slim	0.46/0.45	0.79/0.78	0.69/0.69
v3_wide	0.48/0.47	0.79/0.76	0.65/0.65
v3_slim	0.47/0.48	0.78/0.78	0.71/0.70
v3_sa	0.45/0.44	0.79/0.80	0.76/0.75
v3_sync[1]	0.44/0.43	0.81/0.81	0.87/0.86

Table 1. Loss, accuracy and mAP of the best-epoch models (based on the highest mAP epoch) for the validation and test sets.
1. **v3_sync** uses the augmented loss function defined in (5).

SE	AvgPool2d	Linear	ReLU	Linear	h_sigmoid		
ConvSE	Conv2d	BN	**SE**[1]	ReLU6[2]			
ConvReLU6	Conv2d	BN	ReLU6[2]				
DwS	Conv2d (dw)	BN	ReLU	**SE**[1]	Conv2d (pw)	BN	ReLU
InvRes	Conv2d(1×1)	**DwS**[3]					

Table 2. Basic blocks used in MobileNet-V1-3. **Bold** block components refer to other basic blocks.
1. SE is optional.
2. In MobileNet-V3, ReLU can optionally be replaced with h_swish.
3. ReLU is removed from DwS in this instance.

#	**v1** (59075 parameters)	**v2_slim** (69328)	**v2_wide** (95347)	**v3_slim** (77872)
1	ConvReLU6	ConvReLU6	ConvReLU6	ConvReLU6
2	DwS(c=32)	DwS(c=16)	DwS(c=16)	DwS(c=16, se)
3	DwS(c=64, s=2)	InvRes(t=2, c=24, s=2)	InvRes(t=1.5, c=24, s=2)	InvRes(t=1.5, c=24,s=2)
4	DwS(c=64, s=2)	InvRes(t=1.5, c=24, s=2)	InvRes(t=2, c=32, s=2)	InvRes(t=1.5, c=24,s=2)
5	DwS(c=64, s=2)	InvRes(t=1.5, c=32, s=2)	InvRes(t=2, c=32, s=2)	InvRes(t=2, c=24, s=2)
6	DwS(c=64, s=2)	InvRes(t=2, c=32)	InvRes(t=3, c=32)	InvRes(t=2, c=32, hs)
7	DwS(c=64, s=2)	InvRes(t=2, c=32, s=2)	InvRes(t=3, c=32, s=2)	InvRes(t=2, c=32, s=2, se, hs)
8	DwS(c=64, s=2)	InvRes(t=2, c=32)	InvRes(t=4, c=32)	InvRes(k=5, t=2, c=32, se, hs)
9	AdaptiveAvgPool2d	ConvReLU6(k=1, c=64, p=0)	ConvReLU6(k=1, c=64, p=0)	ConvReLU6(k=1, c=40, p=0, hs)
10		AdaptiveAvgPool2d	AdaptiveAvgPool2d	AdaptiveAvgPool2d
11		ConvReLU6(k=1, c=64, p=0)	ConvReLU6(k=1, c=64, p=0)	ConvReLU6(k=1, c=64, p=0, hs)
12				ConvReLU6(k=1, c=64, p=0, hs)

#	**v3_wide** (108891 parameters)	**v3_sa** and **v3_sync** (58921)	
		Audio	Video
1	ConvReLU6	ConvReLU6	ConvReLU6
2	DwS(c=16, se)	DwS(c=16, se)	DwS(c=16, se)
3	InvRes(t=1.5, c=24, s=2)	InvRes(t=1.5, c=24,s=2)	InvRes(t=1.5,c=24,s=2)
4	InvRes(t=2, c=24, s=2)	InvRes(t=2, c=32, s=2)	InvRes(t=1.5,c=24,s=2)
5	InvRes(t=2, c=32, se, hs)	ConvReLU6(k=1, c=40, p=0, hs)	InvRes(t=2,c=24,s=2)
6	InvRes(k=5, t=2, c=32, s=2, se, hs)	AdaptiveAvgPool2d	InvRes(t=2,c=24,hs)
7	InvRes(k=5, t=3,c=32, se, hs)	ConvReLU6(k=1, c=64, p=0, hs)	InvRes(t=2, c=32, se, hs)
8	ConvReLU6(k=1, c=64, p=0)	ConvReLU6(k=1, c=64, p=0, hs)	InvRes(k=5, t=2, c=32, se, hs)
9	AdaptiveAvgPool2d		ConvReLU6(k=1, c=40, p=0, hs)
10	ConvReLU6(k=1, c=96, p=0)		AdaptiveAvgPool2d
11	ConvReLU6(k=1, c=64, p=0)		ConvReLU6(k=1, c=64, p=0, hs)
12			ConvReLU6(k=1, c=64, p=0, hs)

Table 3. Model architectures. k:=kernel size, default=3; s:=stride, default=1; c:= output channels; t:=expansion ratio; p:=padding, default=$(c-1)/2$; se:=SE used; hs=h_swish used

6. Conclusion

The justifications of MobileNet V1, V2 and V3 in terms of run-time and accuracy were all observed during experimentation, as they behaved in an ascending order of performance. It was also observed that, given the size of the dataset, simpler models (around 60,000 parameters) are most suited. Excessive convolution of the audio input in `v3_slim` was also shown to be an issue, as a halved audio embedding in `v3_sa` performed significantly better than it's heavier symmetric counterpart. Precision of `SPEAKING_AUDIBLE` increased by 10% whilst not only having 75% of the total parameters of `v3_slim`, but also having 154 less parameters and being 28% faster than `v1`. Considering that the V3 model presented in [2] with 2.9 M parameters had latency of 15.8 ms, the real-time application of our structure that is 49 times lighter is beyond doubt. The idea of non-symmetry can also be taken forth in the future with using completely different model types, however the complexity of chosen models will have to taken in to account, as MobileNets are currently state of the art in terms of real-time application [2]. With `v3_sync` the advantage of utilising the notion of synchronisation was evident, as the pre-trained embeddings yielded a higher overall accuracy and precision from the very first epoch. The importance of synchronisation embeddings also justifies the use of a joint loss function, encapsulating both cross entropy and contrastive loss. A further improvement can come from more effective scaling of their α and β contributions to the overall loss in (5).

References

[1] Roth J, Chaudhuri S, Klejch O, Marvin R, Gallagher A, Kaver L, et al. Ava-activespeaker: An audio-visual dataset for active speaker detection. arXiv preprint arXiv:190101342. 2019;.

[2] Howard A, Sandler M, Chu G, Chen LC, Chen B, Tan M, et al. Searching for mobilenetv3. In: Proceedings of the IEEE International Conference on Computer Vision; 2019. p. 1314–1324.

[3] Cour T, Jordan C, Miltsakaki E, Taskar B. Movie/script: Alignment and parsing of video and text transcription. In: European Conference on Computer Vision. Springer; 2008. p. 158–171.

[4] Everingham M, Sivic J, Zisserman A. Taking the bite out of automated naming of characters in TV video. Image and Vision Computing. 2009;27(5):545–559.

[5] Ephrat A, Mosseri I, Lang O, Dekel T, Wilson K, Hassidim A, et al. Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation. arXiv preprint arXiv:180403619. 2018;.

[6] Nagrani A, Chung JS, Zisserman A. Voxceleb: a large-scale speaker identification dataset. arXiv preprint arXiv:170608612. 2017;.

[7] Shillingford B, Assael Y, Hoffman MW, Paine T, Hughes C, Prabhu U, et al. Large-scale visual speech recognition. arXiv preprint arXiv:180705162. 2018;.

[8] Afouras T, Chung JS, Zisserman A. The conversation: Deep audio-visual speech enhancement. arXiv preprint arXiv:180404121. 2018;.

[9] Fisher III JW, Darrell T, Freeman WT, Viola PA. Learning Joint Statistical Models for Audio-Visual Fusion and Segregation. In: Leen TK, Dietterich TG, Tresp V, editors. Advances in Neural Information Processing Systems 13. MIT Press; 2001. p. 772–778. Available from: http://papers.nips.cc/paper/1898-learning-joint-statistical-models-for-audio-visual-fusion-and-segregation.pdf.

[10] Stefanov K, Beskow J, Salvi G. Vision-based active speaker detection in multiparty interaction. In: Grounding Language Understanding GLU2017 August 25, 2017, KTH Royal Institute of Technology, Stockholm, Sweden; 2017. .

[11] Stefanov K, Sugimoto A, Beskow J. Look who's talking: visual identification of the active speaker in multi-party human-robot interaction. In: Proceedings of the 2nd Workshop on Advancements in Social Signal Processing for Multimodal Interaction; 2016. p. 22–27.

[12] Liu Q, Rui Y, Gupta A, Cadiz JJ. Automating camera management for lecture room environments. In: Proceedings of the SIGCHI conference on Human factors in computing systems; 2001. p. 442–449.

[13] Hu Y, Kautz J, Yu Y, Wang W. Speaker-Following Video Subtitles. ACM Trans Multimedia Comput Commun Appl. 2015 Jan;11(2). Available from: https://doi.org/10.1145/2632111.

[14] Anguera X, Bozonnet S, Evans N, Fredouille C, Friedland G, Vinyals O. Speaker diarization: A review of recent research. IEEE Transactions on Audio, Speech, and Language Processing. 2012;20(2):356–370.

[15] Zhang C, Yin P, Rui Y, Cutler R, Viola P. Boosting-based multimodal speaker detection for distributed meetings. In: 2006 IEEE Workshop on Multimedia Signal Processing. IEEE; 2006. p. 86–91.

[16] Chung JS, Zisserman A. Out of time: automated lip sync in the wild. In: Asian conference on computer vision. Springer; 2016. p. 251–263.

[17] Chopra S, Hadsell R, LeCun Y. Learning a similarity metric discriminatively, with application to face verification. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). vol. 1. IEEE; 2005. p. 539–546.

[18] Owens A, Efros AA. Audio-visual scene analysis with self-supervised multisensory features. In: Proceedings of the European Conference on Computer Vision (ECCV); 2018. p. 631–648.

[19] Ren J, Hu Y, Tai YW, Wang C, Xu L, Sun W, et al. Look, listen and learn—a multimodal LSTM for speaker identification. In: Thirtieth AAAI Conference on Artificial Intelligence; 2016. .

[20] Gu C, Sun C, Ross DA, Vondrick C, Pantofaru C, Li Y, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018. p. 6047–6056.

[21] Chung JS. Naver at ActivityNet Challenge 2019–Task B Active Speaker Detection (AVA). arXiv preprint arXiv:190610555. 2019;.

[22] Activity-Net. Task B – Spatio-temporal Action Localization (AVA); 2019. Accessed on 20/03/2020. Available from: http://activity-net.org/challenges/2019/tasks/guest_ava.html.

[23] Brandenburg K. MP3 and AAC Explained; 1999. .

[24] Besacier L, Bergamini C, Vaufreydaz D, Castelli E. The effect of speech and audio compression on speech recognition performance; 2001. p. 301 – 306.

[25] CVD foundation. AVA Actions Dataset host. CVD foundation; 2019. Available from: https://github.com/cvdfoundation/ava-dataset.

[26] OpenCV. Geometric Transformations of Images; 2019. Accessed on 20/03/2020. Available from: https://docs.opencv.org/4.2.0/da/d6e/tutorial_py_geometric_transformations.html.

[27] Watson N. LMDB Memory mapped database; 2019. Available from: https://github.com/jnwatson/py-lmdb.

[28] Arandjelovic R, Zisserman A. Look, listen and learn. In: Proceedings of the IEEE International Conference on Computer Vision; 2017. p. 609–617.

[29] Moattar MH, Homayounpour MM. A review on speaker diarization systems and approaches. Speech Communication. 2012;54(10):1065–1103.

[30] Kinnunen T, Li H. An overview of text-independent speaker recognition: From features to supervectors. Speech communication. 2010;52(1):12–40.

[31] Martinez J, Perez H, Escamilla E, Suzuki MM. Speaker recognition using Mel frequency Cepstral Coefficients (MFCC) and Vector quantization (VQ) techniques. In: CONIELECOMP 2012, 22nd International Conference on Electrical Communications and Computers. IEEE; 2012. p. 248–251.

[32] Friedland G, Vinyals O, Huang Y, Muller C. Prosodic and other long-term features for speaker diarization. IEEE Transactions on Audio, Speech, and Language Processing. 2009;17(5):985–993.

[33] Imseng D, Friedland G. Tuning-robust initialization methods for speaker diarization. IEEE Transactions on Audio, Speech, and Language Processing. 2010;18(8):2028–2037.

[34] Soleymani S, Dabouei A, Iranmanesh SM, Kazemi H, Dawson J, Nasrabadi NM. Prosodic-enhanced siamese convolutional neural networks for cross-device text-independent speaker verification. In: 2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS). IEEE; 2018. p. 1–7.

[35] Chung SW, Chung JS, Kang HG. Perfect Match: Improved Cross-Modal Embeddings for Audio-Visual Synchronisation. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE; 2019. p. 3965–3969.

[36] Arandjelovic R, Zisserman A. Objects that sound. In: Proceedings of the European Conference on Computer Vision (ECCV); 2018. p. 435–451.

[37] Schroff F, Kalenichenko D, Philbin J. Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2015. p. 815–823.

[38] You M, Han X, Xu Y, Li L. Systematic Evaluation of Deep Face Recognition Methods. Neurocomputing. 2020;.

[39] Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:170404861. 2017;.

[40] Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC. Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2018. p. 4510–4520.

[41] tensorflow. MobileNet Transformations of Images; 2019. Accessed on 20/03/2020. Available from: https://github.com/tensorflow/models/blob/master/research/slim/nets/mobilenet/README.md.

[42] Ramachandran P, Zoph B, Le QV. Searching for activation functions. arXiv preprint arXiv:171005941. 2017;.

[43] Hu J, Shen L, Sun G. Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2018. p. 7132–7141.

[44] Yang T, Howard AG, Chen B, Zhang X, Go A, Sze V, et al. NetAdapt: Platform-Aware Neural Network Adaptation for Mobile Applications. CoRR. 2018;abs/1804.03230. Available from: http://arxiv.org/abs/1804.03230.

[45] Loshchilov I, Hutter F. Decoupled Weight Decay Regularization. arXiv preprint arXiv:171105101. 2017;Available from: https://arxiv.org/abs/1711.05101.