The Journal of Supercomputing (2024) 80:16654–16678
https://doi.org/10.1007/s11227-024-06098-6

Automatic phoneme recognition by deep neural networks

Bianca Valéria L. Pereira[1] · Mateus B. F. de Carvalho[1] · Pedro Augusto A. da S. de A. Nava Alves[1] · Paulo Rogerio de A. Ribeiro[1] · Alexandre Cesar M. de Oliveira[1] · Areolino de Almeida Neto[1]

Accepted: 23 March 2024 / Published online: 15 April 2024

Abstract

This work presents a lightweight phoneme recognition model using object detection techniques. This model is mainly proposed to run on devices with low processing power, such as tablets and mobile phones. The use of the combination of hardware network architecture research complemented by the NetAdapt algorithm has led to the use of a simpler and lighter network architecture called MobileNet. The MobileNetV3 convolutional network architecture was combined with the Single-Shot Detection. The databases used in model training were TIMIT and LibriSpeech, both have spoken audios in English. To generate a graphical representation using the audiobases, for each audio, its spectrogram was calculated on the Mel scale. To train the algorithm of phoneme location detection, the temporal position of the occurrence of each phoneme in respective spectrogram is used. Additionally, it was necessary to increase the training dataset, in order to provide improvement in the generalization of the model. Therefore, the two databases were joined and data augmentation techniques were applied to audios. The main idea was to achieve learning using a lightweight architecture that can be used on devices with low processing power, such as tablets and mobile phones. Thus, this research used the MobileNet-Large architecture, which obtained an accuracy of 0.72 mAP@0.5IOU. For comparison, the MobileNet-Small architecture was also used, which obtained an accuracy of 0.63 mAP@0.5IOU.

Keywords Speech recognition · Deep neural networks · Computer vision · Deep learning

Extended author information available on the last page of the article

1 Introduction

The applicability of interacting with machines through audio commands provides greater use of them and expands possibilities of use, entailing greater accessibility for all individuals, especially for those who have physical limitations or disabilities.

The first researches in this area aimed to identify a sound signal and convert it into numerical digits [9]. From this first milestone, several models were implemented. Around the 1980s, some lines of research began to apply the probabilistic theory of hidden Markov models (HMM) in speech recognition systems and gave rise to the first commercial speech recognition systems [21, 35]. Around the 1990s, commercial systems became popular, such as IBM Via Voice and Dragon Systems [3].

More recently, around 2012, a group of researchers proposed a new approach, which consisted of replacing the acoustic model, where a Gaussian process represents this model in most systems, by another one based on deep neural networks (DNNs). This model was used as a feature extractor and was linked to another module based on hidden Markov chains. This change exceeded the performance of the previous state-of-the-art model by more than 30% [17].

After this milestone, several research initiatives have been trying to further improve the performance of current ASR (automatic speech recognition). Some researchers sought to build an end-to-end system based entirely on neural networks. During their research, some groups of researchers obtained satisfactory results by replacing the statistical module, based on HMM and responsible for modeling the sequential structure of speech, by models based on recurrent neural networks (RNNs) [12, 13, 39, 40].

The system output can be either the whole word or the word subdivided into smaller phonetic units, for example, phonemes, syllables, diphones, triphones or even characters. These units are commonly called tokens [2].

Continuous speech recognition systems are systems that commonly present a high complexity when compared to others. This is mainly because in human speech there are usually no pauses among tokens. These systems have been gradually replaced by models that use deep neural networks [12, 13, 39, 40].

The main objective of this work was to implement an approach for recognizing phonemes in words. Graphical representations of audio were used, to which pattern recognition and object detection techniques were applied using deep neural networks. The contributions of this work include the use of a simplified neural network architecture, which provided results with satisfactory accuracy using little processing power when compared to more robust networks. It was therefore possible to use the proposed model in devices with limited processing power, such as smartwatches and smartphones.

2 Bibliographic review

Some works in the literature have obtained promising results by applying deep learning techniques to raw audio signals without any kind of pre-processing [4, 31, 32].

However, the use of pre-processing resources usually enhances the extraction of representative features, which are relevant to the pattern recognition process. Therefore, the pre-processing drives the efforts of the model for the understanding of the really useful information and refraining from irrelevant information, such as background noises or unexpressive spectra.

With this in mind, recent works divide the audio recognition task into two modules, the first one seeks to elaborate an acoustic model and the second one is a classifier. The first module is responsible for extracting representative features of the audio spectrum. These features are subsequently sent to the second module, which should correlate the information in order to identify the analyzed sequences.

Meftah et al. [30] showed a study analyzing some of the main feature extractors used in the literature, using an HMM-based model as a classifier. They used Linear Predictive Coding (LPC), Mel-Frequency Cepstral Coefficients (MFCC), Perceptual Linear Prediction (PLP), Logarithmic Mel-Filter Bank Coefficients (FBANK), Mel-Filter Bank Coefficients (MELSPEC) and Linear Prediction Reflection Coefficients (LPREFC) as the sound representation for their analysis. This study is done using a base of phonemes in the Arabic language. Grozdic et al. [14] used Deep Denoising Autoencoders (DDAE) for whispered speech recognition. DDAE is applied for generating whisper-robust cepstral features. Three types of cepstral coefficients were used in those experiments: MFCC, TECC (Teager-Energy Cepstral Coefficients) and TEMFCC (Teager-based Mel-Frequency Cepstral Coefficients). This system was tested and compared in terms of word recognition accuracy with conventional hidden Markov model (HMM) speech recognizer in an isolated word recognition task with a real database of whispered speech (WhiSpe).

Several works have proposed acoustic models using the combination of some transformation techniques and deep learning techniques. In the work of Quintanilha et al. [34], CNN and RNNs were used. The audio signal of a speech excerpt is transformed into MFCCs and used as the system input. They used the Brazilian Portuguese speech dataset (BRSD) which was built by combining 4 datasets from different sources (LapsBM, CSLU: Spoltech Brazilian Portuguese, Voxforge and Sid dataset).

In the work of Glackin et al. [10] demonstrated how CNN, which is known for state-of-the-art performance for image processing tasks, can be adapted to learn the acoustic model (AM) component of an ASR system. They used spectrograms as input and phonemes as output classes for training to perform speech recognition. For this, they chose to use the TIMIT corpus to train the acoustic model because it has phonemic transcription.

Fan and Liu [8] showed one approach combining CNNs and RNNs and the presented a comparison among audio inputs transformed into Mel-scale spectrogram and FBANKs. Sree and Vijaya [42] proposed to enhance the power of DBN further by pre-training the neural network using particle swarm optimization (PSO). Three

Table 1 The classifier and objective proposed in the aforementioned works

Authors' work	Classifier/learning algorithm	Objective
[30]	HMM-based model	ASR
[14]	Deep Denoising Autoencoders (DDAE)	ASR
[34]	CNN and RNNs	ASR
[10]	CNN-AM	ASR
[8]	CNNs and LSTM-CTC	ASR
[42]	The basic PSO, SGPSO and NMPSO are applied in pre-training the DBN	ASR
[15]	CNN, LSTM, GRU and ensemble learning (MLPC)	Recognizing emotions
[43]	Neural network called an information quantizer (IQ)	ASR

Table 2 The dataset used in the aforementioned studies

Authors' work	Dataset properties
[30]	A corpus of Arabic speech was created based on selected recordings of recitations from the Holy Quran
[14]	WhiSpe
[34]	BRSD
[10]	TIMIT corpus
[8]	TIMIT corpus
[42]	The speech corpus Kazhangiyam
[15]	RAVDESS dataset
[43]	TIMIT corpus and Mboshi

variations of PSO namely, the basic PSO, second generation PSO (SGPSO) and the new model PSO (NMPSO) are applied in pre-training the DBN to analyze their performance on phoneme classification.

Gupta et al. [15] presented the analysis and classification of speech spectrograms for recognizing emotions in RAVDESS dataset. Feature extraction from speech utterances is performed using Mel-Frequency Cepstrum Coefficient. Thereafter, deep neural networks are employed to classify speech into six emotions (happy, sad, neutral, calm, disgust and fear). The paper puted forward an analysis of Bag of Visual Words that uses speeded-up robust features (SURF) to cluster them using K-means and further classify them using support vector machine (SVM) into aforementioned emotions. Out of the five DNNs deployed, (i) Long Short-Term Memory (LSTM) on MFCC and, (ii) Multilayer Perceptron (MLP) classifier on MFCC outperform others, giving an accuracy score of 0.70 (in both cases). Also, it achieves a precision score between 0.77 and 0.88 for the classification of six emotions.

Finally, Wang et al. [43] bridged the gap between the linguistic and statistical definition of phonemes and proposed a novel neural discrete representation learning model for self-supervised learning of phoneme inventory with raw speech and word labels. Under mild assumptions, Wang et al. [43] proved that the phoneme inventory

Table 3 The representation of the sound used as input in the aforementioned studies

Authors' work	Sound representation
[30]	LPC, MFCC, PLP, FBANK, MELSPEC and LPREFC
[14]	MFCC, TECC and TEMFCC
[34]	MFCCs
[10]	Spectrograms
[8]	Mel-scale spectrograms and FBANKs
[42]	Discrete wavelet transform (DWT) applied on the wave form repeatedly to extract six levels of DWT features
[15]	Spectrograms - Feature extraction from speech utterances is performed using MFCCs
[43]	Frame-level phoneme transcriptions

learned by they approach converges to the true one with an exponentially low error rate. Moreover, in experiments on TIMIT and Mboshi benchmarks, your approach consistently learns a better phoneme-level representation and achieved a lower error rate in a zero-resource phoneme recognition task than previous state-of-the-art self-supervised representation learning algorithms.

Tables 1, 2 and 3 summarize the classifier, the objective, the dataset used and the representation of the sound that served as input for the learning algorithm in the works mentioned above.

The various challenges constantly proposed to the community are factors of fundamental importance. These challenges aim to stimulate the development of new solutions that can accurately detect a wide range of different object classes. As examples of these challenges, one can mention Google AI Open Images Object Detection Track 2018, PASCAL Visual Object Classes Challenge 2007 and 2012 (VOC2007, VOC2012), Microsoft COCO: Common Objects in Context (MS COCO), ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) [5, 7, 23, 26, 38].

As a result of the computer vision community's efforts to overcome these challenges, some very interesting tools have emerged, such as You Only Look Once (YOLO) [36] and Single-Shot MultiBox Detector (SSD) [27].

The first steps to investigate the application of object detection techniques to classify phonemes in audio signal spectrograms were taken by [1]. In their work, the authors proposed the use of phoneme recognition using spectrograms of the English TIMIT base and an Arabic base. They used the YOLO architecture and also an architecture called CenterNet, in both of which they obtained results comparable to recently published articles on phoneme recognition.

Table 4 The classifier and objective proposed in the aforementioned works

Authors' work	Architectures	Dataset	Sound representation	Objective
[1]	YOLO and CenterNet	TIMIT and an Arabic base	Spectrograms	ASR
Ours	SSD and MobileNet	TIMIT and LibriSpeech	Spectrograms	ASR

In our work, we proposed the use of the SSD object detection system using the MobileNetV3 neural network as a backbone. We used an approach similar to that of [1]; however, the proposal was to use a leaner and more simplified architecture. As a result of this approach, it was possible to obtain results with satisfactory accuracy using modest computational resources, compared to more complex networks. The TIMIT and LibriSpeech English-language audio databases were used to train the phoneme recognition system.

Table 4 summarizes the two proposals, mentioning the classifier, the objective, the dataset used and the representation of the sound that served as input for the learning algorithm in the aforementioned works.

In [1]'s work, the focus was general because he used the state of the art in object detection. In our work, the focus was on a more simplified architecture with the aim of obtaining good results without the need for a large amount of computing power, as this would make it possible to apply the system to a device with less computing power, such as an embedded system or a smartphone.

3 Voice processing

The voice is an inherently human characteristic based on the production of articulated sounds, which are represented by a language. Speech is an analog, continuous and non-periodic signal. Due to technical limitations, digital computers do not allow the voice signal to be stored considering an infinite set of points, so periodic sampling must be performed. The necessary quantitative foundation for this purpose is provided by the sampling theorem [24].

After recording and storing the signal digitally, to get a good representation of the signal, a sliding window is used. This process consists of sliding along the signal in a window, whose length is usually limited to between 20 and 40 ms. To avoid loss of information, the distance between a window and its successor is usually smaller than the total width of the window, so the windows overlap.

For each window extracted, the Short-Time Fourier Transform (STFT) was applied. With the result of the previous step, the spectral power of the signal is calculated. The goal of this procedure is to intensify the amounts of energy in different frequency bands.

The process of transforming the power spectrum to the so-called Mel scale is a method that models the frequency response similar to what occurs in the human auditory system [8, 11]. To transform the power spectrum to the Mel scale, a bank of K filters must be applied to the spectral power $S(k)$. The filter bank is composed of filters spaced according to the Mel frequency scale, represented by Eq. 1:

$$\text{Mel}(f) = 1125 \times ln\left(1 + \frac{f}{700}\right) \tag{1}$$

where f is the frequency of the signal.

The effect of using the Mel-scale spectrogram is to promote spacing among the low magnitude frequencies located at the bottom of the spectrogram, while the high magnitude frequencies are slightly compressed.

4 Convolutional network architectures

Nowadays, many network architectures are available for image recognition. Some prominent examples are the Google AI Open Images Object Detection Track 2018, PASCAL Visual Object Classes Challenge 2007 and 2012 (VOC2007, VOC2012), Microsoft COCO: Common Objects in Context (MS COCO), ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) challenges [5, 7, 23, 26, 38].

Around 2012, convolutional networks started to be widely spread in the computer vision area, when Alex Krizhevsky won the ImageNet challenge (ILSVRC 2012) [38] with his AlexNet network architecture [22]. Since then, several network architectures have been proposed. The modifications are very diverse, such as increasing the number of layers or neurons. In general, the larger the network and the more sophisticated techniques used, the higher the accuracy; however, this efficiency is not reflected in relation to the size of the model and its processing speed.

In real-world applications, where a high-performance machine is not always available, more sophisticated models are difficult to be executed in a device with computational restrictions. Even when a datacenter can be used to compensate this restriction, in some situations, there is no internet access.

The relationship between accuracy and efficiency of architectures is an important factor in a project involving neural networks. However, the demand for networks that work offline on low computational capacity devices has caused some smaller architectures such as MobileNet, ShuffleNet and SqueezeNet to emerge [19, 20, 44].

The reduction of image size through pooling technique or using a stride value greater than one is a real possibility. These techniques can reduce the total number of parameters to be processed.

Convolutional networks perform many mathematical operations, therefore, demand a high processing power. In order to overcome it, a technique called pointwise convolution or 1×1 convolution was developed [25].

The computational cost Θ_{standard} of the standard convolution operation is represented by Eq. 2:

$$\Theta_{\text{standard}} = D_k \cdot D_k \cdot M \cdot N \cdot D_f \cdot D_f \tag{2}$$

where D_k represents the size of the filter, M and N represent the number of input channels and the number of output channels, respectively, and D_f represents the dimensions of the original image.

In order to reduce the computational cost Θ_{standard}, the Depthwise Separable Convolution technique emerged. This technique divides the previous process into two steps. The first step performs a convolution similar to the previous one, each channel is treated individually. The difference is in the end of the process, where the initial number of channels is preserved and the operation is applied only once, thus, Eq. 3 is obtained.

$$\Theta_{\text{Depthwise}} = D_k \cdot D_k \cdot M \cdot D_f \cdot D_f \tag{3}$$

where $\Theta_{\text{Depthwise}}$ is the computational cost of the Depthwise Separable Convolution.

Depthwise convolution is more efficient compared to standard convolution; however, this step is only responsible for applying the filters to the channels and does not combine them to extract new features. Therefore, a second step is required that computes a linear combination among the previous (Pointwise convolution 1×1 or Pointwise convolution). At this point, a $1 \times 1 \times M$ filter is used. The result of this second step is a feature map that has $D_f \times D_f \times 1$ dimension, so for N maps, we obtain $D_f \times D_f \times N$ dimension. The combination between the Depthwise convolution and the Pointwise convolution 1×1 has the computational cost Θ_{dp} described by Eq. 4.

$$\Theta_{\text{dp}} = D_k \cdot D_k \cdot M \cdot D_f \cdot D_f + N \cdot M \cdot D_f \cdot D_f \tag{4}$$

The understanding of this operation is not intuitive and many times, the first impression hides its effectiveness or utility; however, this technique is extremely powerful to condense the information along the channels of the image inputs or filters in a given layer. The MobileNet model uses this this technique. The MobileNet architecture is currently in its third version. In order to improve the architecture, the authors proposed a convolutional block that combines the pointwise and depthwise convolutional techniques used in the first version with the residual connection techniques [16].

The intention is to provide an architecture with low density layers. In order to obtain a good accuracy without increasing the data density, the authors presented a convolution block called Bottleneck Residual Block.

The function of this block is to reduce the amount of data that are passing through the network, as the name suggests, the block generates a "bottleneck" of information in its output. This set of convolutions consists of three stages: the first one applies a pointwise convolution layer to the received data. Next, there is a depthwise layer, which performs a filtering of the expanded data. Finally, in the last part of the block, a pointwise layer "compacts" the data and reduces its dimensionality. In addition to the operations mentioned above, the block has a residual connection between the first and last stages [18, 41].

4.1 Object detection

The goal of the conventional classification process, for example, using convolutional networks, is to extract features through the first layers and at the end of the process the last layers of the network are responsible for performing a classification. In object detection task, however, at the end of the process, this task indicates not only the object classification, but also its location in the image. Therefore, there are two output layers: the first one classifies a certain object, while the second output layer indicates the location of this object.

There are two widespread methods for object detection in images, one based on sliding windows and the other based on region proposals. The first mentioned method is commonly called a one-step detector, while the second one is called a two-step detector. As the name suggests, the one-step detector performs the detection in only one stage. The two-step detectors, on the other hand, first need to use some method to propose regions, and then, they perform the detections. The two-stage detectors have high accuracy rates considering the location and also the classification, while the one-stage detectors stand out in the detection speed. The most representative example in the literature of a two-stage detector is the Faster R-CNN [37]. As examples of one-stage detectors, we can mention the systems You Only Look Once (YOLO) and Single-Shot Multibox Detector (SSD) [27, 36].

4.1.1 Single-shot multibox detector

According to the number of objects to be detected, a collection of delimiters, called bounding boxes, is created. For each delimiter, an estimate of presence of one object is produced. After detections, the non-maximum suppression algorithm runs aiming to discard irrelevant bounding boxes.

This detector has some peculiarities that provide versatility and agility in detection, without compromising the accuracy of its predictions. Among the features of that detector, one can highlight the addition of some extra layers that are coupled to the base architecture used. These extra layers follow a pattern in which their size is progressively reduced, in order to provide detection at multiple scales, as shown in Fig. 1.

Some extra layers added progressively decrease in size to assist the detection. One can observe that all outputs of the layers, both from the base architecture, as well as the outputs of the extra layers, are connected directly in parallel with the detection layer. After the detection layer, the non-maximum suppression step is applied [27, 36]. According to the authors, the goal of this method is to reduce overfitting. Besides, the authors proposed a "by-pass" among the layers connecting them directly to the classifier at the end of the network.

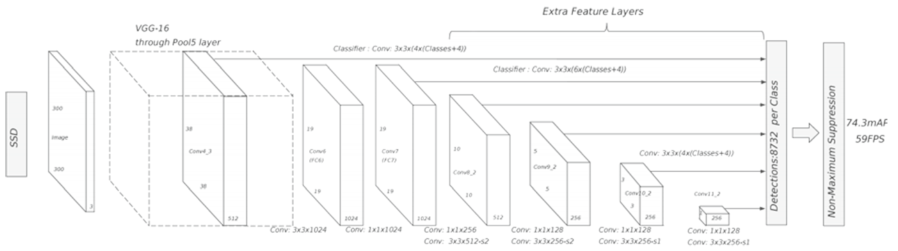


Fig. 1 Example of a VGG-16 architecture as the base backbone. Extra layers with reduced size were added to assist the detection. All outputs of the layers are directly connected (black arrows) with the detection layer. Later, the non-maximum suppression limitates the bounding boxes with lower scores [27]

During the training of the SSD system, it is necessary to determine which bounding boxes are closer to the annotation of the object's real position and train the network accordingly. For each prediction, the set of standard bounding boxes is varied in order to diversify aspects related to its shape, location and the bounding box scale so as to cover objects in different positions. Therefore, for each of the delimiters in the standard set of bounding boxes, the intersection over union index (IoU) is computed. For this, the annotation that corresponds to the location in the image of the original object is used and the prediction that has the highest index is selected. The intersection over the union is also called Jaccard's index and can be expressed by Eq. 5:

$$J(A,B) = \frac{A \cap B}{A \cup B} \tag{5}$$

where A and B are sets of images.

According to the author, the objective function used in the SSD system is derived from the MultiBox system [6]; however, the function in the SSD was optimized to handle multiple categories. The objective function of this system is composed of the localization loss (loc) and the confidence loss (conf), according to Eq. 8.

The localization loss (loc) is responsible for correcting the error between the original annotation position and the position predicted by the network through the bounding box. For that, only positive combinations are considered. Be X_{ij}^{p} an indicator for the equivalence between the i-th bounding box and the j-th annotation of a class p in image. If IoU is greater than 0.5, $X_{ij}^{p} = 1$, otherwise, $X_{ij}^{p} = 0$. The localization loss is the same Smooth L1 loss used in YOLO [36], according to Eq. 6:

$$L_{\text{loc}}(X,l,g) = \sum_{i \epsilon \text{Pos}}^{N} \sum_{m \epsilon c_x, c_y, w, h} X_{ij}^{p} \text{smooth}_{L1}(l_i^m - \hat{g}_j^m) \tag{6}$$

where N is the number of matched default boxes, the parameter l represents the marking predicted by the network and parameter g represents the original marking on the image, where $\hat{g}_j^{c_x} = \frac{cx_{g_j} - cx_{d_i}}{w_{d_i}}$, $\hat{g}_j^{c_y} = \frac{cy_{g_j} - cy_{d_i}}{h_{d_i}}$, $\hat{g}_j^{w} = \log\left(\frac{w_{g_j}}{w_{g_i}}\right)$ and $\hat{g}_j^{h} = \log\left(\frac{h_{g_j}}{h_{g_i}}\right)$, while parameters cx_{g_j} and cy_{g_j} correspond, respectively, the x and y position of the center of the bounding box g_j, the parameters cx_{d_i} and cy_{d_i} correspond, respectively, the x and y position of the center of the bounding box d_i, the parameters w_{d_i}, w_{g_j} and w_{g_i} correspond, respectively, the width of the bounding box d_i, g_j and g_i, the parameters h_{d_i}, h_{g_j} and h_{g_i} correspond, respectively, the height of the bounding box d_i, g_j and g_i.

The confidence loss (conf) metric represents the error calculated when making a prediction. For each positive combination, the objective function is adjusted according to the evaluation score for that respective class. As for the negative combinations, the zero class is penalized, which corresponds to the "background" detection in the image. For this, one should apply the softmax function considering the scores obtained in the classification according to Eq. 7:

$$L_{\text{conf}}(X,c) = -\sum_{i\epsilon \text{Pos}}^{N} X_{ij}^{p} \log(\hat{c}_i^p) - \sum_{i\epsilon \text{Neg}} \log(\hat{c}_i^0) \tag{7}$$

where c is class score, $\hat{c}_i^p = \frac{e^{c_i^p}}{\sum_p (e^{c_i^p})}$ and N is the total number of positive combinations.

The overall objective function is described by Eq. 8:

$$L(X,c,l,g) = \frac{1}{N}(L_{\text{conf}}(X,c) + \alpha L_{\text{loc}}(X,l,g)) \tag{8}$$

where N is the number of bounding boxes representing a positive combination and the parameter α is a weight to be multiplied by the localization loss.

5 Methodology

Concerns the materials and methods employed in this work. The first ones consist basically of databases and a framework. The methods used here were phonetic division, data augmentation and metrics.

5.1 Databases

Two databases were used: TIMIT and LibriSpeech. TIMIT is an English-language audio database designed to provide data for acoustic-phonetic studies and to assist in the development and evaluation of automatic speech recognition systems. The base consists of audios from 630 speakers, covering the eight major dialects of American English. Each speaker recorded ten phonetically rich sentences, generating a total of 6300 sentences, or 5.4 h. The utterances were recorded using 16-bit resolution and frequencies range up to 16 kHz. For each audio-recorded utterance, there are orthographic, phonetic and word transcriptions, which all have temporal annotation.

LibriSpeech is a free database built from the audiobooks made available in the LibriVox project [33]. It contains about 1000 h of speech with frequencies in the 16 kHz range. The files are divided into three subsets, with sizes of 100, 360 and 500 h. Unlike the TIMIT database, the annotations in this database were automatically verified using alignment and speaker identification algorithms.

This base originally does not have phonetic alignment. Therefore, to be able to use this base in this work, it was necessary to use a tool to force phonetic alignment [29]. This algorithm has some pre-trained templates for certain languages. The files containing the information regarding the phonetic and word annotations were obtained from the work of [28]. For more details about the alignment algorithm, see the mentioned works. In this work, we chose to use the subset of the base that contains 100 h of recordings. Even so, only a part of the recordings was selected to complement the sample bases used during training.

5.2 Framework

In order to develop this project, the Google Collab framework was used, as well as, the API's TensorFlow and Keras. To perform experiments in this work, we chose to use the SSD object detector in combination with the MobileNet network architecture. This was decided in order to have a low processing cost and to work in a device with low computational capacity or offline.

It is worth remembering that this algorithm was previously developed in another work [18] and was applied to the ImageNet classification challenge [38]. In this work, we took advantage of this architecture applied to the task of phoneme recognition.

As mentioned in Sect. 4, the MobileNet architecture is responsible for performing the convolutional part of the process and has two versions: MobileNetV3-Large and MobileNetV3-Small. The large version consists of the complete architecture of the network and the small one is a reduced version, which uses a smaller amount of parameters.

5.3 Phoneme division

Originally the audios were recorded containing entire sentences. The base contains annotations of the temporal position of each spoken word and phoneme throughout each recording. Thus, the sentences were divided into isolated phonemes. Furthermore, it also provides faster detection, as it uses smaller units for detection, which provide a faster response when compared to recognition of complete utterances.

The audiobases used in this work, for each audio sample, there is a text document containing annotations regarding the temporal position of each phoneme present in the utterance. A program was developed that receives the temporal information of the beginning and end of each phoneme and separates the audios into phonemes or words and organizes them according to their phonetic classification.

When converting the audio format to spectrogram, phonetic annotation was also performed taking into account the position of phonemes in pixels in the spectrogram image. Figure 2 illustrates the constitution of phonetic annotations on the word "covers." For each audio sample, a spectrogram and a markup file were created containing information on the position of each phoneme in that image. Thus, one can

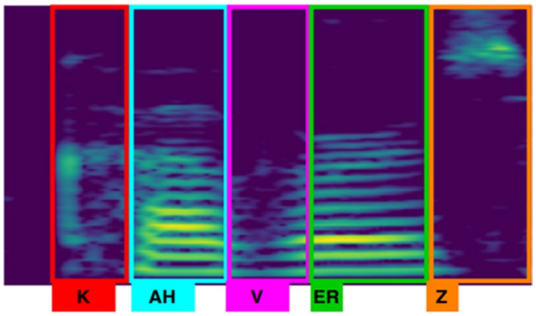


Fig. 2 Example of phonetic annotation on the spectrogram of an audio recording of the word "covers"

visualize the markings through software used to annotate images, such as LabelImg. Through the markings file, we facilitated the process of converting the images and the annotation process for the standard used during the training of the object detection algorithms.

5.4 Data augmentation

Most processes involving the use of deep network algorithms require an extensive number of samples for training, and these samples should be so diverse as possible. However, many times the process of obtaining data is very costly. Thus, one way to reduce costs in obtaining data is through data augmentation techniques. To deal with conventional images, it is very common to apply image deformation techniques such as: stretching or shrinking their size, rotating them clockwise or counterclockwise, enlarging or shrinking the images, modifying the intensity and color balance, and even removing a random piece of the images.

However, as the images in this work correspond to a distribution of signals over time, some techniques cannot be applied. For example, mirroring horizontally the image may totally invert the meaning of the pronunciation of a word. Procedures to stretch or shrink the image can be applied in this context since its effect on audio is similar to procedures of increasing or decreasing the recording speed.

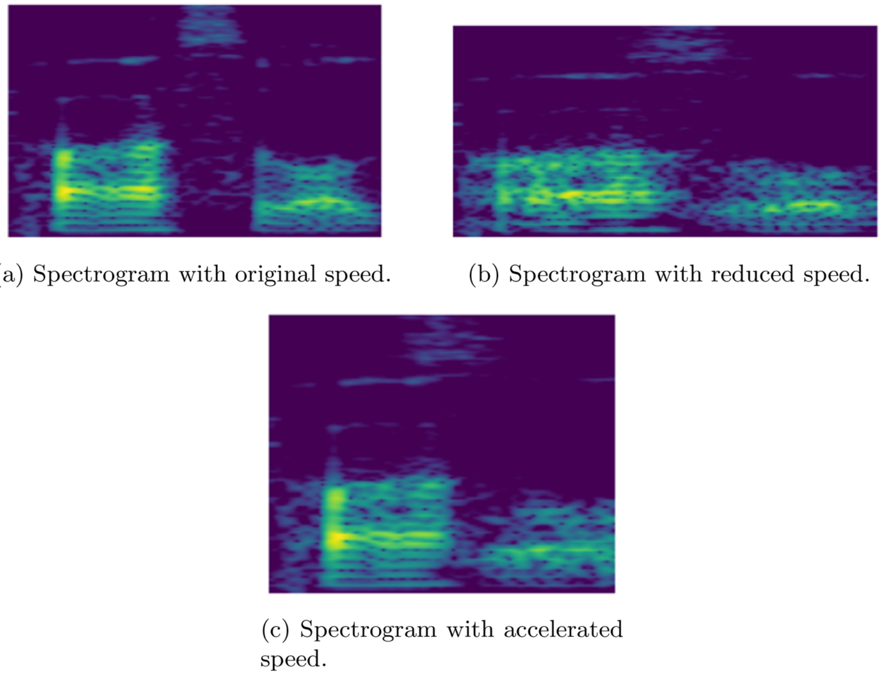
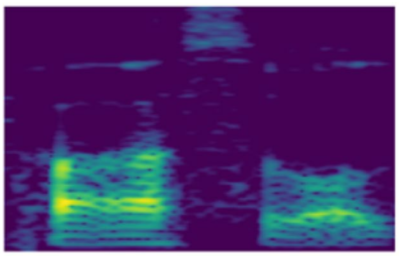

(a) Spectrogram with original speed.
(b) Spectrogram with reduced speed.
(c) Spectrogram with accelerated speed.

Fig. 3 Effect on the spectrogram of audios recorded with variations in speed


(a) Original spectrogram.

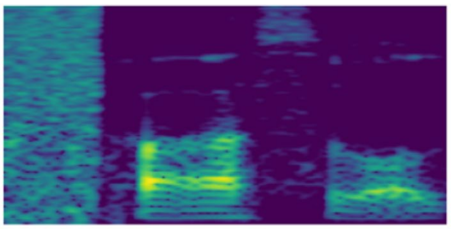

(b) Spectrogram with audio displaced from the source.

Fig. 4 Displacing the audio by adding noise before the original audio's position

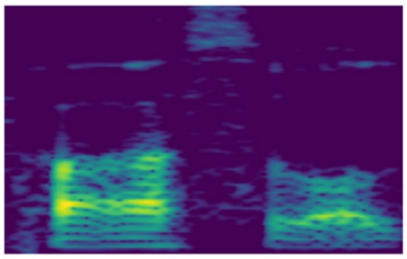
In Fig. 3, one can observe the effect of changes in the spectrogram as a function of increasing and decreasing the recording speed of the original audio. Figure 3a shows the spectrogram of audio at normal speech rate. In Fig. 3b, the image has been stretched on the horizontal axis to represent audio with reduced speech rate. In Fig. 3c, the opposite comes to light, i.e., the speed is faster and the speaking time is shorter.

Another technique used was the displacement of the audio of the original recording by adding a noise excerpt at the beginning of the audio, as shown in Fig. 4. Thus, the corresponding information to the speech is time delayed by a few seconds.

Figures 4a, b are spectrograms of the word "possible." While Fig. 4a illustrates the spectrogram of the audio in its original form, Fig. 4 shows the effect of adding a noise section at the beginning of the recording in order to displace it in time. Figure 4b shows a light spot at the beginning (that would be the noise) and then the original audio spectrogram equal to Fig. 4a.

Noise was also applied throughout the original recording. To do this, one can generate a white noise and multiply it by a coefficient, and then add it to the original recording. Besides adding a white noise, one can also use another type of noise.

Figure 5 illustrates the application of white noise in an audio recording that contains the word "possible." While Fig. 5a illustrates the audio's spectrogram in its


(a) Original spectrogram.

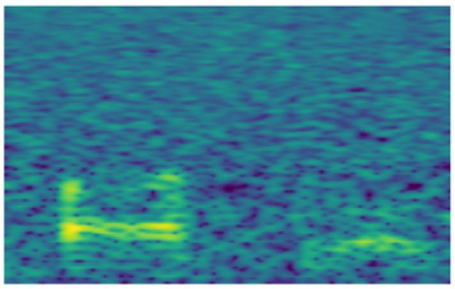
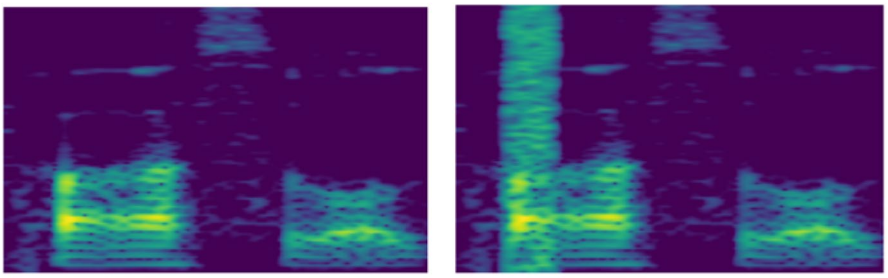

(b) Spectrogram with white noise added.

Fig. 5 Technique that consists of adding a bank of white noise throughout the audio recording



(a) Original spectrogram. (b) Spectrogram with localized noise added.

Fig. 6 Technique for adding localized noise to certain parts of audio recordings

original form, Fig. 5b shows the effect of adding white noise across the entire length of the audio.

One can see in Fig. 5b that the light spot that appears at the beginning of Fig. 4b extends across the entire spectrogram image, but it is still noticeable that the audio of the word is present in the spectrum, the stronger green spots are noticeable in the same locations in Figs. 5a, b.

Another technique used was the addition of localized noise in certain passages of the audio recordings. In this case, according to the position of phonemes in the audio and its duration, a localized noise was applied in order to disfigure one of the phonemes as illustrated in Fig. 6. Figures 6a, b are spectrograms of the word "possible."

While Fig. 6a illustrates the audio spectrogram in its original form, Fig. 6b shows the effect of adding noise located at a random location in the audio. The difference among the spectrograms is a vertical spot on the left side of Fig. 6b, which would be the localized noise to disfigure the phoneme.

5.5 Metrics

Two metrics were used: the Phone Error Rate and the Mean Average Precision.

5.5.1 Phone error rate

The Phone Error Rate (PER) metric is similar to the Word Error Rate (WER) metric. Both are metrics widely used to evaluate the performance of speech recognition or translation systems. The PER metric considers four important parameters to measure the performance of the recognition system, namely substitutions (S), deletions (D) and insertions (I), divided by the total number of predictions (N), according to Eq. 9.

$$\mathrm{PER} = \frac{S + D + I}{N} \tag{9}$$

5.5.2 Mean average precision

Mean Average Precision (mAP) is a widely used metric in object detection. This metric is calculated through the intersection over union (IoU) of the areas of the object's real marking in the image and the marking predicted by the algorithm during classification. In general, the detection of the object location is correct if the IoU is greater or equal to 0.5. If a detection occurs and IoU is greater than or equal to 0.5, the detection is True Positive (TP), otherwise, False Positive (FP). If there is no detection, the model failed to detect the object and thus the detection is classified as False Negative (FN). The mAP is the average model accuracy considering a given confidence level, represented by IoU. To calculate the average precision of the model, Eq. 10 is used.

$$\mathrm{AP} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{10}$$

To calculate the mAP, the average of the accuracies (AP) of all classes is considered, according to Eq. 11:

$$\mathrm{mAP} = \frac{1}{N} \sum_{j=1}^{N} \mathrm{AP}_j \tag{11}$$

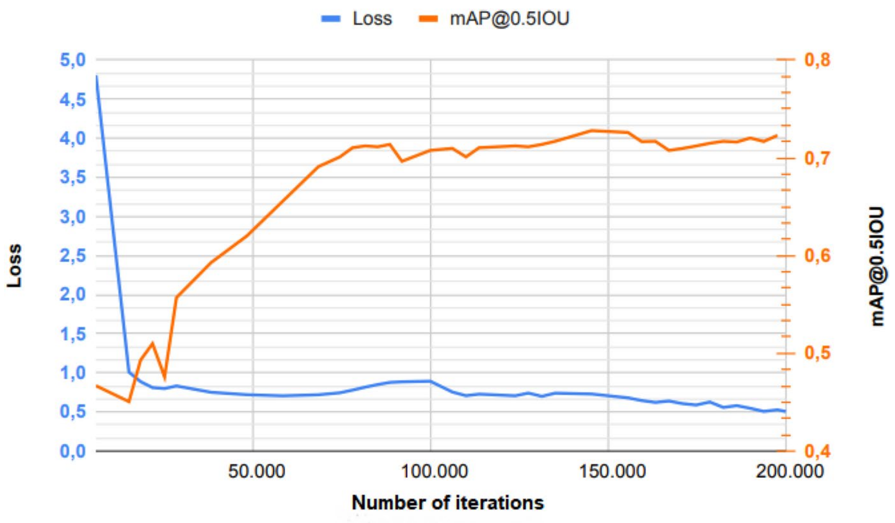
6 Results

The use of data augmentation techniques, presented in Sect. 5, individually generated different effects during training, and the impacts were verified during the tests. Even so, during the experiments, we tried to combine the data augmentation techniques in order to provide a greater generalization of the model and thus ensure a satisfactory result.

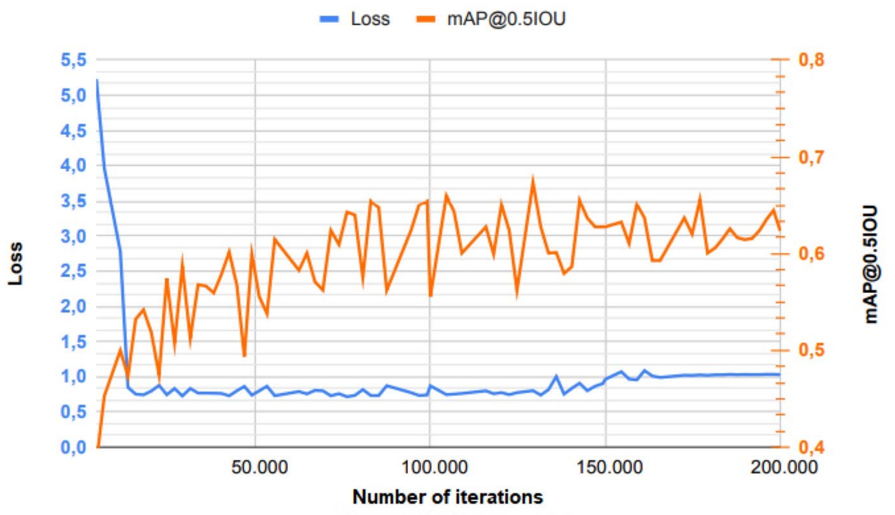
In addition to data augmentation techniques, parts of another audio database, LibriSpeech, were also used. During the experiments, it was decided to use only a subset of this database, in order to verify its impact on training. The aggregation of these new samples allowed to obtain a satisfactory result.

For phoneme recognition, k-fold cross-validation was applied, where the input data was divided into five data subsets, i.e., the percentage of the dataset assigned is 80% for training, 10% for validation and 10% for testing. The samples generated by the data augmentation techniques remained in the training dataset.

In Fig. 7, one can verify the progress of one of the system trainings. During this training, it was used a batch size equal to 16, an initial learning rate of 0.1 and a momentum term equal to 0.9. In this training, the combinations of data augmentation techniques (local noise, displacement and 20% speed change) applied to the TIMIT database training data were used, and a subset of the LibriSpeech database was added to increase the number of samples in the database and favor the network generalization.



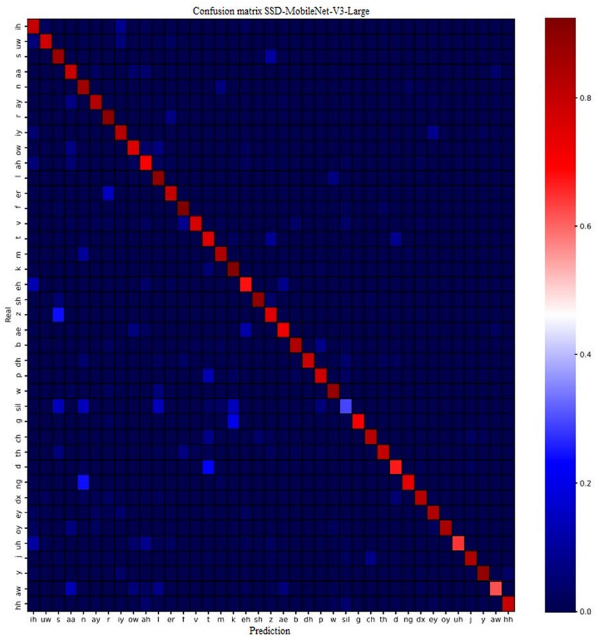
(a) Cost and efficiency curve of the MobilenetV3-Large.



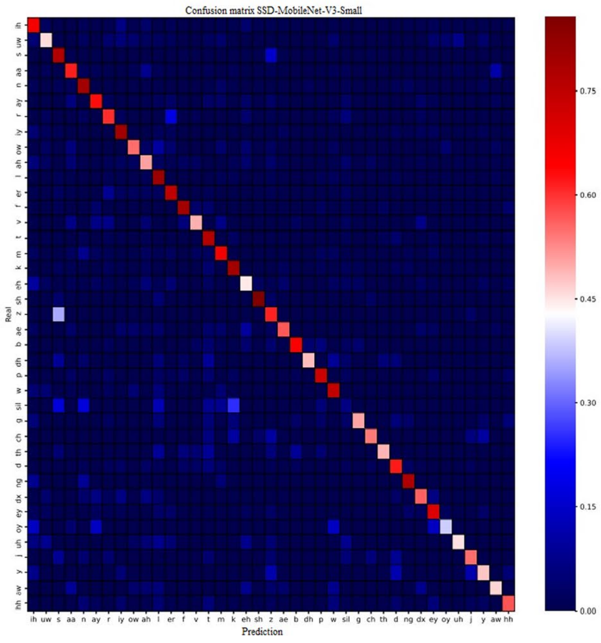
(b) Cost and efficiency curve of the MobilenetV3-Small.

Fig. 7 Graph of net training evolution for 200,000 iterations

In both graphs presented in Fig. 7, the markings on the left side of the graph represent the variation of the training Loss, while the right side shows the scale corresponding to the accuracy measured by mAP. Likewise, the blue line represents the variation of Loss during the training, while the orange line corresponds to accuracy measurement.



(a) Confusion matrix containing the 39 phoneme classes of the model using the Large architecture.



(b) Confusion matrix containing the 39 phoneme classes of the model using the Small architecture.

Fig. 8 Confusion matrix graph of networks trained for 200,000 iterations

The graph shown in Fig. 7a represents the training of the full version of the network, the MobileNetV3-Large architecture. The training of this version occurred in a low-noise way, presenting remarkable progress and stability. Among the first 10 thousand iterations, there was a significant decrease in the Loss parameter, while accuracy showed significant growth until around the 80 thousandth iteration. At the end of the process, the accuracy presented a value of 0.72 mAP, while the Loss finished near 0.5.

On the other hand, the graph shown in Fig. 7b represents the MobileNetV3-Small architecture training. This training, in turn, presented a high level of noise, with many oscillations throughout the process. This characteristic expressed a certain difficulty of the system to learn the data patterns. In this case, it happened mainly due to the reduced parameters of the architecture, a factor that impaired the learning task. At the end of the process, the accuracy presented a value of 0.63 mAP, while Loss finished near 1.0.

Figure 8a illustrates the confusion matrix of the best model obtained throughout the experiments. The closer to the dark red, the higher the hit rate. Analyzing the figure, for example, for the class *z*, we have a red square, so it was recognized. We can also notice that class *z* was sometimes recognized as class *s* and *t* for presenting a lighter blue than the other classes. The class sil was the only one that was identified a few times correctly when compared to others, because line it is the only one that presents a light blue color on the diagonal line, while all the other classes are red. Some classes were identified as other classes, for example, class *k* was identified as *g* and sil at some moments, so the blue of these two classes is lighter than the other classes.

One can verify that most predictions are in accordance with the corresponding classes, except the class sil. As previously commented, the system considered most of the occurrences of this class as "background" of the image, so it disregarded its classification. Considering this possibility, an alternative to circumvent this problem would be to eliminate the class of silence, since it can be considered background of the image, and discarding it would not cause losses to the model.

Figure 8b illustrates the confusion matrix of the Small model. Since this model is smaller and less robust than the model previously presented, it has less effective results. Different from the confusion matrix illustrated in 8a, there are some gaps in the main diagonal. The classes uw, ah, v, eh, dh, g, th, uh, y and aw show a faint red or white color on the diagonal line, this means that about 0.45 of the total were correctly classified. The classes sil and oy have been correctly classified very few times, so they show a blue color on the diagonal line. As the other classes have been correctly classified several times, they show a strong red color on the diagonal. Even though there are many inaccurate predictions, analyzing the bic picture, the model presented above average results.

In Fig. 9, a circular graph is illustrated with the objective of exposing the performance in relation to the mAP considering the classes individually. It was considered the intersection index over union (IoU) equal to 0.5. Regarding the more robust network, it can be verified that there was a good performance in most classes, except for the class "sil," which corresponds to the pauses and the silences during the audio signal. It was verified that this problem occurred because this class is often confused

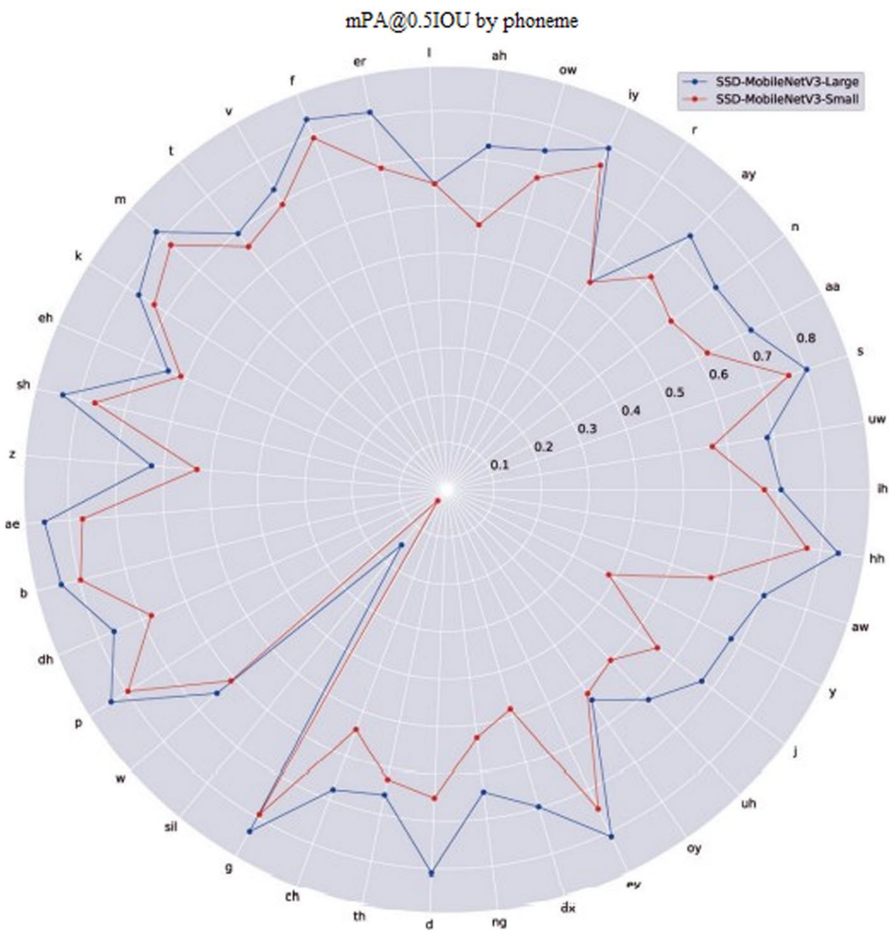


Fig. 9 Circular graph comparing the mAP for each phoneme class of the Large and Small models

Table 5 Results obtained during the testing phase for the MobileNetV3-Large and -Small models using test data from the TIMIT base

Model	mAP(0.5IOU)	PER(%)	Parameters	Average time
SSD-MobileNetV3-Large	0.729	19.47	5.4M	9–10 h
SSD-MobileNetV3-Small	0.632	31.02	2.5M	3–5 h

with the "ackground" of the image. According to the methodology adopted by the object detection system, the detections considered as "background" of the image are simply discarded to avoid disturbing the classifier during the recognition task.

Table 6 Results obtained by [1]

Model	mAP (0.5IOU)	PÈR(%)	Parameters	Average time
YOLOv3	0.785	16.34	65.9M	Not informed by the authors
CenterNet	0.766	15.89	62.2M	Not informed by the authores
YOLO V3 Tiny	0,68	25.57	5.5M	Not informed by the authores

6.1 Comparison of results

The results obtained during the experiments of this work are presented in Table 5. A comparison was made using the two models of the same architecture used, the full version (large) and the reduced version (small).

Notably, the complete network architecture obtained better results. This occurs because it has more attributes and consequently greater generalization power. On the other hand, the amount of parameters used by the reduced network is less than half of the amount used by the larger network.

Table 6 shows the results obtained in the work of [1], which uses a methodology similar to the one used in this work. The results are comparable because they were obtained using the same database during the testing phase. By analyzing the tables, the product of this work presented slightly lower results compared to [1]. However, when comparing the number of parameters from both systems, the ones used in this work, although presenting lower accuracy, require an expressively lower amount of parameters. Being a system that uses fewer parameters, the training is faster, less memory is used, consequently less computational resources are used and this is very appropriate and advantageous for systems with low computational power, such as tablet, cell phones and smartwatches.

When comparing the YOLOv3 architecture to the MobileNetV3-Large SSD, there is a difference of 0.056 in accuracy, but it becomes minimal when compared to the availability of computational resources due to the significant reduction in the number of parameters. YOLOv3 uses twelve times the number of parameters of the MobileNetV3-Large SSD, so its computational cost is much higher. It is not useful for systems with low computational power to have one thousand one hundred percent more parameters to gain only 6% in accuracy. When comparing the Yolo V3 Tiny architecture to the SSD-MobileNetV3-Large, the number of parameters is equivalent, but the results obtained by the SSD-MobileNetV3-Large are superior.

7 Conclusion

This work comprises the area of signal processing and pattern identification to phoneme classification using techniques of computer vision. In order to represent the signal graphically, using spectrograms calculated based on the audio signals.

Throughout the work, the need to increase the amount of data for training was identified. Generally, acquiring data is a costly task, so some data augmentation techniques have been applied like increasing and decreasing the recording speed of the original audio, displacing the audio by adding noise before the original audio's position, among others.

Over the past few years, several object detection algorithms have been proposed and many of them present satisfactory results. Nevertheless, besides the proper functioning of the algorithm in terms of accuracy of its predictions, other important factors are the latency of response of the predictions and the processing capacity required for the operation of these systems.

As a result of this work, the developed model presented satisfactory results regarding phoneme identification. The model offers an efficient accuracy, even using reduced number of parameters, for instance 5.4M for the MobileNet-Large model and 2.5M for the MobileNet-Small model. From the results, one can apply the algorithms for object recognition in spectrograms. In mobile devices, which have low processing capacity. This will allow the use of the model in a free, unlimited and standalone way, without the need to use an external service that is often requires payments for the number of requests requested.

In the present work, we sought to build a phoneme classifier, but nothing prevents us from using the same strategies for the word detection task, since the database itself contains annotations referring to words. To do this, it would be enough to select a certain set of words to train the algorithm in a similar way to that performed in this work. Although the word detection approach generates a model with limited vocabulary, depending on the desired application, this type of detection can be useful.

In addition to word detection, one can apply the same technique to recognize other kinds of sound elements. To do this, simply use databases with temporal annotations of the occurrence of sound effects to enable the training of the algorithm. MFCC, FBanks or some geometric representation such as vectors, linear combination of vectors, among others, or a union of these techniques.

Furthermore, one can modify the network backbone and use other architectures in conjunction with the SSD system or even use another object detection system such as YOLO or Faster R-CNN.

Acknowledgements The authors express their gratitude to Federal University of Maranhão, specially to InovTec Lab.

Funding This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

1. Algabri M, Mathkour H, Bencherif MA et al (2020) Towards deep object detection techniques for phoneme recognition. IEEE Access 8:54663–54680
2. Bresolin AdA (2008) Speech recognition through units smaller than the word, using wavelet packet and svm, in a new hierarchical decision structure. Ph.D. thesis, Federal University of Rio Grande do Norte
3. Coniam D (1999) Voice recognition software accuracy with second language speakers of English. System 27(1):49–64

4. Dai W, Dai C, Qu S et al (2017) Very deep convolutional neural networks for raw waveforms. In: 2017 IEEE International Conference on Acoustics. Speech and Signal Processing ICASSP. IEEE, pp 421–425
5. Deng J, Dong W, Socher R et al (2009) Imagenet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, pp 248–255
6. Erhan D, Szegedy C, Toshev A et al (2014) Scalable object detection using deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 2147–2154
7. Everingham M, Van Gool L, Williams CK et al (2010) The Pascal visual object classes VOC challenge. Int J Comput Vis 88(2):303–338
8. Fan R, Liu G (2018) CNN-based audio front end processing on speech recognition. In: 2018 International Conference on Audio, Language and Image Processing ICALIP. IEEE, pp 349–354
9. Furui S (1981) Cepstral analysis technique for automatic speaker verification. IEEE Trans Acoust Speech Signal Process 29(2):254–272
10. Glackin C, Wall JA, Chollet G et al (2018) Convolutional neural networks for phoneme recognition. In: ICPRAM, pp 190–195
11. Gordillo CDA (2013) Recognition of continuous voice combining MFCC and PNCC attributes with SS, WD, map and FRN robustness methods. Ph.D. thesis, Pontifical Catholic University of Rio de Janeiro
12. Graves A, Fernández S, Gomez F et al (2006) Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd International Conference on Machine Learning, pp 369–376
13. Graves A, Mohamed Ar, Hinton G (2013) Speech recognition with deep recurrent neural networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, pp 6645–6649
14. Grozdic DT, Jovicic S, Subotić M (2017) Whispered speech recognition using deep denoising autoencoder. Eng Appl Artif Intell 59:15–22
15. Gupta V, Juyal S, Hu YC (2022) Understanding human emotions through speech spectrograms using deep neural network. J Supercomput 78:1–30
16. He K, Zhang X, Ren S et al (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 770–778
17. Hinton G, Deng L, Yu D et al (2012) Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag 29(6):82–97
18. Howard A, Sandler M, Chu G et al (2019) Searching for mobilenetv3. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 1314–1324
19. Howard AG, Zhu M, Chen B et al (2017) Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861
20. Iandola FN, Han S, Moskewicz MW et al (2016) Squeezenet: alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. arXiv preprint arXiv:1602.07360
21. Juang BH, Levinson S, Sondhi M (1986) Maximum likelihood estimation for multivariate mixture observations of Markov chains Corresp. IEEE Trans Inf Theory 32(2):307–309
22. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. Adv Neural Inf Process Syst 25:1097–1105
23. Kuznetsova A, Rom H, Alldrin N et al (2018) The open images dataset v4: unified image classification, object detection, and visual relationship detection at scale. arXiv e-prints pp arXiv–1811
24. Lathi BP (2006) Linear signals and systems-2. Bookman
25. Lin M, Chen Q, Yan S (2013) Network in network. arXiv preprint arXiv:1312.4400
26. Lin TY, Maire M, Belongie S et al (2014) Microsoft coco: common objects in context. In: European Conference on Computer Vision. Springer, pp 740–755
27. Liu W, Anguelov D, Erhan D et al (2016) Ssd: single shot multibox detector. In: European Conference on Computer Vision. pringer, pp 21–37
28. Lugosch L, Ravanelli M, Ignoto P et al (2019) Speech model pre-training for end-to-end spoken language understanding. arXiv preprint arXiv:1904.03670
29. McAuliffe M, Socolof M, Mihuc S et al (2017) Montreal forced aligner: trainable text-speech alignment using kaldi. In: Interspeech, pp 498–502
30. Meftah A, Alotaibi YA, Selouani SA (2016) A comparative study of different speech features for Arabic phonemes classification. In: 2016 European Modelling Symposium EMS. EEE, pp 47–52

31. Muckenhirn H, Doss MM, Marcell S (2018) Towards directly modeling raw speech signal for speaker verification using CNNs. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP. IEEE, pp 4884–4888
32. Palaz D, Doss MM, Collobert R (2015) Convolutional neural networks-based continuous speech recognition using raw speech signal. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP. IEEE, pp 4295–4299
33. Panayotov V, Chen G, Povey D et al (2015) Librispeech: an ASR corpus based on public domain audio books. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP. IEEE, pp 5206–5210
34. Quintanilha IM, Biscainho LWP, Netto SL (2017) Towards an end-to-end speech recognizer for portuguese using deep neural networks. In: Proceedings of 35th Simpósio Brasileiro de Telecomunicaçãues e Processamento de Sinais
35. Rabiner L (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77(2):257–286. https://doi.org/10.1109/5.18626
36. Redmon J, Divvala S, Girshick R et al (2016) You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition CVPR
37. Ren S, He K, Girshick R et al (2015) Faster R-CNN: towards real-time object detection with region proposal networks. Adv Neural Inf Process Syst 28:91–99
38. Russakovsky O, Deng J, Su H et al (2015) Imagenet large scale visual recognition challenge. Int J Comput Vis 115(3):211–252
39. Sak H, Senior AW, Beaufays F (2014) Long short-term memory recurrent neural network architectures for large scale acoustic modeling. Google
40. Sak H, Vinyals O, Heigold G et al (2014) Sequence discriminative distributed training of long short-term memory recurrent neural networks. Google
41. Sandler M, Howard A, Zhu M et al (2018) Mobilenetv2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 4510–4520
42. Sree BL, Vijaya M (2020) Building acoustic model for phoneme recognition using PSO-DBN. Int J Bus Intell Data Min 16(4):506–523
43. Wang L, Feng S, Hasegawa-Johnson M et al (2022) Self-supervised semantic-driven phoneme discovery for zero-resource speech recognition. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) pp 8027–8047. https://doi.org/10.18653/v1/2022.acl-long.553, https://aclanthology.org/2022.acl-long.553
44. Zhang X, Zhou X, Lin M et al (2018) Shufflenet: an extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Elsevier, pp 6848–6856

Authors and Affiliations

Bianca Valéria L. Pereira[1] · Mateus B. F. de Carvalho[1] · Pedro Augusto A. da S. de A. Nava Alves[1] · Paulo Rogerio de A. Ribeiro[1] · Alexandre Cesar M. de Oliveira[1] · Areolino de Almeida Neto[1]

✉ Bianca Valéria L. Pereira
lopes.bianca@discente.ufma.br

Mateus B. F. de Carvalho
mateus.barros@discente.ufma.br

Pedro Augusto A. da S. de A. Nava Alves
augusto.pedro@discente.ufma.br

Paulo Rogerio de A. Ribeiro
paulo.ribeiro@ecp.ufma.br

Alexandre Cesar M. de Oliveira
alexandre.cesar@ufma.br

Areolino de Almeida Neto
areolino.neto@ufma.br

1 Federal University of Maranhão (UFMA), Av. dos Portugueses, 1966 - Vila Bacanga, São Luís, Maranhão 65080-805, Brazil