End-to-End Speech Recognition with Raw Audio Singals and 1D Convolutions

Juan Pablo Herrera Alvarez
University of Technology Sydney
Sydney, Australia
12869246@student.uts.edu.aa

***Abstract*— Convolutional neural networks are a powerful form of neural networks. Their coverage ranges from classifying images and identifying multiple objects in a video feed, to writing or responding to real-time conversations. Most of these convolutional networks are several layers deep and use data in a matrix form. This paper focuses on a different kind of shallow neural network that uses one-dimensional array-shaped data to build a simple speech recognition system that classifies raw audio files into 30 different categories without requiring additional acoustic or language models. The system consists of a combination of 1D convolutions with fully connected dense layers and data augmentation in the form of noise injection. Our study shows that 1D convolutions are suitable for the feature extraction of raw audio signals and combined with fully connected layers, it can achieve a competent validation accuracy considering the low footprint and hardware requirements.**

Keywords*— speech recognition, convolutional neural networks, raw audio signal, audio data augmentation.

I. Introduction (*Heading 1*)

Traditional automatic speech recognition (ASR) pipelines require a significant amount of infrastructure in terms of acoustic and language models, as well as the division of the problem into several subtasks to achieve the high levels of accuracy. On the other hand, recent improvements in machine learning have made it possible to train systems in an end-to-end manner, where every step of the process co-occurs, and no acoustic or language models are needed [1].

State of the art end-to-end alternatives using deep recurrent neural networks require a significant amount of computing power and are usually trained using multiple GPUs and require hundreds of hours for refining the model architecture [2]. Bearing in mind the fact that at the time of writing this document and training our model we did not have access to either a dedicated GPU or hundreds of hours for refining the model architecture, we will not be showing state-of-the-art results. Instead, we will be presenting the reader with a lightweight approach to the problem that consists of a combination of 1D convolution layers alongside fully connected layers and can be replicated with a consumer-grade computer. The rest of the paper is organised as follows. In section 2, we introduce our CNN approach and its different architectures. In section 3, we present different results. In section 4, we discuss the results and the performance of our model. We conclude the paper in section 5.

II. Model Architecture

As stated by Palaz et al., despite the success achieved by spectral based acoustic features for speech recognition problems, using raw speech signals for modelling has proved to be effective and achieved a comparable performance, and even surpass its counterparts under more "noisy" conditions [3]. In our specific problem, we will refer to raw audio signals as the audio files on which the only pre-process applied to the data involves resampling the file from the original sample rate to a shorter one[1].

A. Case for CNNs

As mentioned in the introduction of this document, we will be making use of Convolutional Neural Networks (CNNs) in our modelling process. In particular, we will use them for the feature extraction or filter stage of our models.

CNNs main three characteristics include locality, weight sharing, and pooling. In a speech recognition context (and others) these characteristics are desirable and can help to obtain better results when compared to more traditional deep neural networks (DNNs) alternatives. A clear explanation of why these characteristics are important can be found in [4]; however, in this document, we use the explanation presented in [5][6]. In their articles, Zhang et al. and Sainath et al. explain that CNNs explicitly exploit structural locality in the feature space and because of weight sharing and pooling properties, CNNs will give the model better spectral and temporal invariance. Low-footprint CNNs have also proven to be more successful in speech recognition problems when compared to their DNNs counterparts [7].

1) 1D Convolution

Considering that we will be working with raw audio files, we need a way to use CNNs in our problem. The answer that we have found consists of the usage of 1D Convolutions. The difference between 1D convolutions and more traditional 2D convolutions is that, unlike its pair, this layer creates a convolution kernel that is convolved with the layer input over a single spatial (or temporal) dimension to produce a tensor of outputs. In the figure below, the reader can appreciate the basic functioning of a 1D convolutional layer. In the example, the Conv1D layer has a number of filters equal to 10 and a kernel size of 80. As shown, the filter moves along the temporal axis by a quantity indicated by the "stride"

[1] The sample rate is measured in **Hertz** (Hz) and means "samples per second". In our particular case, we used a target sample rate of 8,000Hz, mainly because that is the end of the spectrum where most people are capable of hearing clearly.

hyperparameter and computes the different weights for the samples contained inside the filter space.

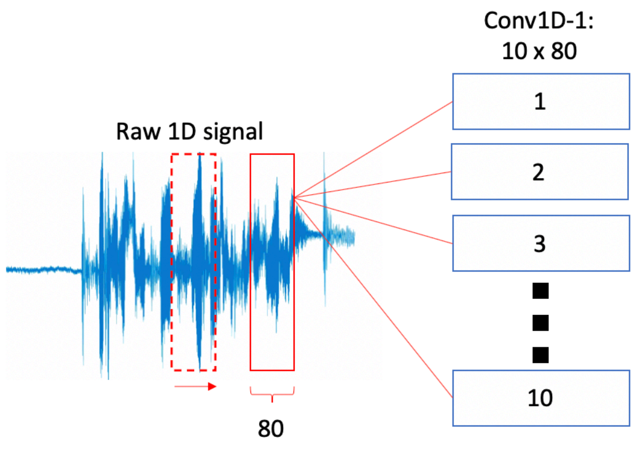

Fig. 1. Representation of a 1D convolutional layer

While the 2D convolutions intent is to operate on two-dimensional data such as images and videos, 1D convolutions operate in one-dimensional data such as audio or time series. 1D convolutions have proven to be more effective than 2D convolutions and are thus preferred in our case [8].

Some of the advantages of 1D convolutions over 2D convolutions are listed below:

- 1D convolutions rely on simple array-based operations rather than matrix-based operations as in 2D convolutions, making them computational light weight.
- 1D convolutions require only shallow architectures while 2D convolutions require deep architectures for the same task.
- 2D convolutions might require expensive hardware for training, while 1D convolutions can operate efficiently on traditional at home hardware.

The above advantages of 1D convolutions compared to 2D convolutions make them and ideal choice for training our network speech recognition.

B. *Batch Normalisation*

When reading different sources looking for inspiration for our work, we have found how, in most applications of speech recognition related problems, Batch normalisation is the norm. Upon further inspecting, we have also found recent research that suggests that batch normalisation not only helps to speed up the training process but can also further improve the generalisation capabilities of the models [9].

Batch normalisation focuses on normalising the activations in the hidden layers of the neural networks by transforming the distribution of the data to one with a zero-mean and constant standard deviation. This technique extends to the hidden/intermediate layers of the network. In their original paper on batch normalisation, Ioffe and Szegedy highlight that batch normalisation may reduce internal covariate shift, which is the drifting of the distribution of activations at the time of training, in turn, affecting the inputs passed to the subsequent layers in the network [10].

C. *MaxPooling*

The importance of pooling layers, as explained by Abdel-Hamid [4] comes from the fact that pooling operation excels in handling small frequency shifts that are common in speech signals. Since we are working with 1D convolutions, we need to use a pooling method that is compatible with that approach. Max-pooling works by creating position invariance on large local regions. It then down samples the input audio signal. Thus, max pooling improves the convergence rate by selecting relevant invariant features, thereby improving the generalization of the model [11].

As stated by Scherer et al., the purpose of pooling layers is to achieve spatial invariance by reducing the resolution of the feature maps. In our particular case, the units combine the input from a patch of units of size n. This window can be of arbitrary, and they can be overlapping. The max-pooling function

$$a_j = \max_N(a_i^n u(n))$$

applies a window function $u(n)$ to the input patch and computes the maximum in the neighbourhood. Then, the resulting feature map has a lower resolution [12].

D. *Data Augmentation*

Neural networks are highly data reliant to avoid overfitting. However, many of the processes and operations that have a need for deep learning techniques do not hold sufficient amounts for data for training their models. In such scenarios, data augmentation techniques provide multiple tools that can scale up the dataset without compromising the quality of the inherent data [13].

To increase the amount of data our model is exposed to, we have opted for the inclusion of data augmentation in the form of noise injection. The noise injection process consists of slightly modifying the samples in our audio files by adding a random number multiplied by a noise coefficient. The resulting samples, while different from the originals, still contain the essential features required for classification and most importantly, remain "audible". In the following figure, one can observe an example of the word happy with and without background noise[2].

[2] We also considered another approach for data augmentation in the form of modification of the pitch; however, the modelling process did not produce satisfactory results, and we opted for excluding it from the analysis.

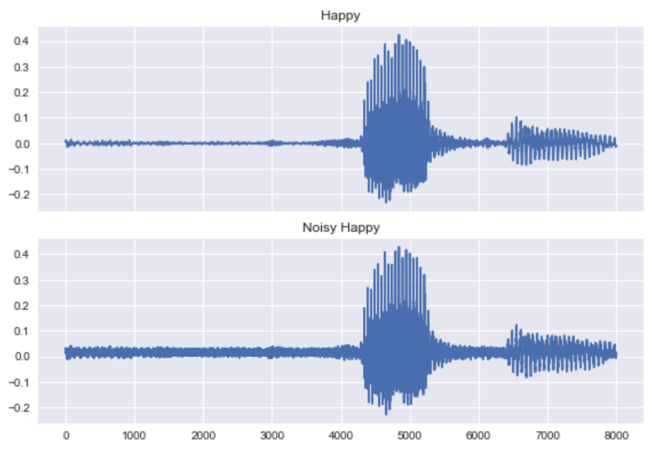

Fig. 2. One sample of the word happy is selected for illustration; the original sample corresponds to the word happy after resampling to 8,000Hz only. In the second image, random noise is added to the sample to generate an effect that can be described as background noise.

E. *Model Construction*

In general, we can describe our model as two-step approach in which on a first stage, the model takes the inputs and pass them through a series of convolution layers; we will refer to this as the feature extraction stage. At the extraction stage, the model deploys a series of four one-dimensional convolutional layers. The number of filters in the convolutional layers range from 8 to 32 in steps of 8. Further adding a one-dimensional convolutional layer with increased filter sizes could potentially improve the model performance; however, it would be at the trade-off of training time. In terms of the size of the kernels for each layer, it ranges between 100 and 20 respectively from layer 1 to 4.

The max-pooling layers have a filter size of 2 each. Considering the form of the data in processing audio signals that range from maximum to minimum, average-pooling was also experimented as a choice. However, as no significant improvement was noted on the performance of the model, max-pooling layers were retained for the final model.

On the second part, the model takes the outputs of the feature extraction stage and pass them through a more traditional DNN architecture that consists of three dense layers including classification layer and batch normalisations; we will refer to this as the classification stage. The network also has dropout layers, that filter out 30% of the data, to help improve the efficiency of training times and reduce the chance of overfitting on training data. The choice of metric for model performance evaluation on the validation data is 'accuracy'. 'Adam' is the preferred choice of optimiser for the model. Since the data for the model is a multi-class data set, that requires prediction of categories amongst 30, categorical cross-entropy is chosen to be the loss component of the model.

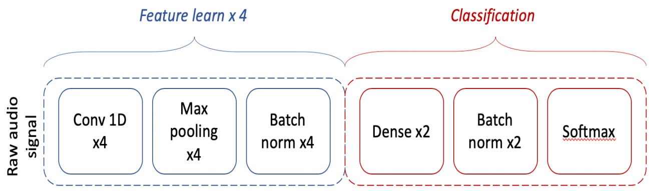

Fig. 3. In the feature extraction stage, four iterations of convolution/max-pooling/batch-normalization are considered. The classification stage includes two dense hidden layers as well as the a dense layer with SoftMax activation function for classification.

III. Experiments and Results

The dataset used for training and validation of our model correspond to the TensorFlow speech recognition challenge in Kaggle and contains 64,721 different audio files divided into 30 different word classes.

Starting from a simple model with a single 1D convolution layer and one dense layer for classification, we can observe how the validation accuracy achieved by this approach reaches 21.91%. We have decided to take this progressive approach to build our model mainly because we wanted to avoid overfitting and also as part of the experimenting process.

On a second iteration of the model and now understanding that 1D convolutions are indeed relevant for speech recognition problems, we proceed to add two extra convolutions and increase the number of filters in both of them. The new model yields a validation accuracy of 51.12%. In a third iteration of the model, we add two extra dense layers on top of the convolutions; the system can now reach validation accuracy of 65.89%. Looking at the time that it takes to complete each training cycle/epoch, one can notice how this extra accuracy that one can obtain by adding the extra dense layers can become very expensive in terms of training accuracy at a high rate. At this point, we stopped modifying the general architecture of the model and started focusing the attention on other ways to increase the system performance.

As explained in a previous section, the concept of data augmentation can be of tremendous help to increase both the amount and variability of the data to which our models are exposed to. The model "Extra Conv1D Augmented V1" is the direct result of using the exact same model architecture as in "Extra Conv1D" but using an augmented dataset for training resulting in a validation accuracy of 76.38%. After including the concept of stratification, adding one extra 1D convolutional layer, and re-training our model using a new dataset, we managed to increase the validation accuracy to 79.34%.

In the table below, the reader can observe the results for the different models trained during the analysis.

TABLE I. Modelling Results

Model	Validation accuracy	Time per epoch
Single Conv1D	21.91%	11s
Multi Conv1D	51.12%	27s
Extra Conv1D	65.89%	64s
Extra Conv1D Augmented V1	76.38%	57s
Extra Conv1D Augmented V2	74.40%	41s
Extra Conv1D Augmented V3	79.34%	51s

Fig. 4. Multiple iterations of the model are built in a progressive manner.

IV. Discussion

In the previous table, the reader can observe not only the evolution of our experiments with the architecture but also the evolution of the accuracy results. However, since the previous representation does not consider how the training process was, there are a few interesting results that are worth mentioning. To that end we will be examining the Accuracy-Loss plots, for the model "Extra Conv1D Augmented V1" and "Extra Conv1D Augmented V3".

In figure 5, one can appreciate the results for the “Extra Conv1D Augmented V1” model. By looking at the accuracy plot, the reader should be able to observe how even though, we reported an accuracy of 76.38% on the validation set; this result is not necessarily indicating a good fit. The reason, is that when one looks at the evolution of the validation accuracy, one can observe how it is not a stable process and the complete process shows significant differences for the accuracy metric in-between epochs. This is clear evidence that there is an imbalance between our train set and our validation set. Therefore, our validation set cannot be considered representative of the train set.

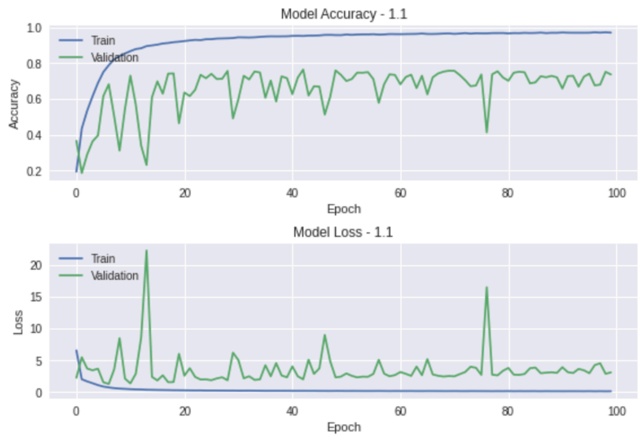

Fig. 5. Accuracy-Loss plot for the “Extra Conv1D Augmented V1” model.

With the previously mentioned imbalance in mind, we proceeded to generate new train and validation sets using the class membership vector for stratification. The results of that process correspond to the model “Extra Conv1D Augmented V2”; however, as reported in the Table I, the results of that process yielded an accuracy of 74.4%. With the previous outcome in mind, we proceeded to train a new model once again , using the balanced train-validation set but including an extra convolution layer. The results for the new training process, as observed in the figure 6, produce a model less volatile and more consistent that yields a final validation accuracy of 79.34%.

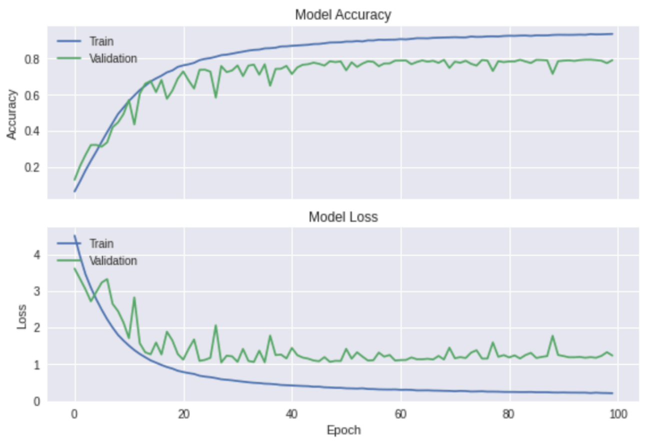

Fig. 6. Accuracy-Loss plot for the “Extra Conv1D Augmented V3” model. The system architecture includes and extra convolution layers making the total number of convolutions before the classification stage equals to four.

V. Conclusion

In this work, we propose a lightweight system that can be trained under two hours and produces a validation accuracy that consistently is just under 80%. The model architecture can be described as shallow and straightforward as opposed to deep and complex since we only make use of four convolution layers and three dense layers, including the last layer for classification. However, those characteristics correspond to trade-offs that we were willing to make since at the current stage we do not have the necessary resources to work with such big dataset in a way that allows faster training and therefore the necessary number of iterations that will allow further improvements. The model presented in this document should be seen as proof of concept were, we show that it is possible to work with raw audio files and obtain decent results. In order to improve the system performance, there are several actions that can be employed including the usage of recurrent neural networks (RNNs) and increasing the number of layers of our model considering different layers such as ConvLSTM layers, look ahead convolutions or residual neural networks (ResNet).

References

[1] Palaz, D., Doss, M.M. and Collobert, R., 2015, April. Convolutional neural networks-based continuous speech recognition using raw speech signal. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4295-4299). IEEE.

[2] Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A. and Ng, A.Y., 2014. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*.

[3] Palaz, D., Collobert, R. and Doss, M.M., 2013. Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks. *arXiv preprint arXiv:1304.1018*.

[4] Abdel-Hamid, O., Mohamed, A.R., Jiang, H., Deng, L., Penn, G. and Yu, D., 2014. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, *22*(10), pp.1533-1545.

[5] Zhang, Y., Chan, W. and Jaitly, N., 2017, March. Very deep convolutional networks for end-to-end speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4845-4849). IEEE.

[6] Sainath, T.N., Kingsbury, B., Mohamed, A.R., Dahl, G.E., Saon, G., Soltau, H., Beran, T., Aravkin, A.Y. and Ramabhadran, B., 2013, December. Improvements to deep convolutional neural networks for LVCSR. In *2013 IEEE workshop on automatic speech recognition and understanding* (pp. 315-320). IEEE.

[7] Huang, J.T., Li, J. and Gong, Y., 2015, April. An analysis of convolutional neural networks for speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4989-4993). IEEE.

[8] Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M. and Inman, D.J., 2019. 1D convolutional neural networks and applications: A survey. *arXiv preprint arXiv:1905.03554*.

[9] Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G. and Chen, J., 2016, June. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning* (pp. 173-182).

[10] Bjorck, N., Gomes, C.P., Selman, B. and Weinberger, K.Q., 2018. Understanding batch normalization. In *Advances in Neural Information Processing Systems* (pp. 7694-7705).

[11] Nagi, J., Ducatelle, F., Di Caro, G.A., Cireşan, D., Meier, U., Giusti, A., Nagi, F., Schmidhuber, J. and Gambardella, L.M., 2011, November. Max-pooling convolutional neural networks for vision-based hand gesture recognition. In *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)* (pp. 342-347). IEEE.

[12] Scherer, D., Müller, A. and Behnke, S., 2010, September. Evaluation of pooling operations in convolutional architectures for object recognition. In *International conference on artificial neural networks* (pp. 92-101). Springer, Berlin, Heidelberg.

[13] Shorten, C. and Khoshgoftaar, T.M., 2019. A survey on image data augmentation for deep learning. *Journal of Big Data*, *6*(1), p.60.