Speaker Differentiation Using A Convolutional Autoencoder

Mohamed Asni
School of Electrical Engineering and Computer Science
University of Ottawa, Lemay.ai
Ottawa, Canada
mohamed@lemay.ai

Daniel Shapiro
School of Electrical Engineering and Computer Science
University of Ottawa, Lemay.ai
Ottawa, Canada
daniel@lemay.ai

Miodrag Bolic
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Canada
mbolic@uottawa.ca

Tony Mathew
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Canada
tmath043@uottawa.ca

Leor Grebler
UCIC
Toronto, Canada
leor@ucic.io

***Abstract*—In this work, a deep learning solution for differentiating speaker voices in audio given two microphone sources is presented as a step towards solving the cocktail party problem. A convolutional autoencoder was trained using a small sample size of data to associate audio snippets with categorical labels. Audio snippets collected as part of this work were used for training and evaluating the model. Audio was converted to mel-frequency cepstrum representation prior to classification. The collective processed data was labeled according to the person or collection of persons speaking. The model was trained and evaluated using data with two, three, four, five, and six categories. The result was a model that recognizes when different people are speaking in a 2-person, 3-person, 4-person, 5-person, and 6-person conversation with an accuracy of 99.29%, 97.62%, 96.43%, 93.43%, and 88.1%, respectively. Experimental comparisons between the five versions of the model are presented.**

***Index Terms*—deep learning, convoluional neural network, voice separation, signal processing**

I. Introduction

This paper outlines the research, design, and implementation of a deep learning solution for differentiating speaker voices in audio originating from two microphones simultaneously. Prior art, such as [1], focused on using a deep learning solution to perform speaker separation given a mixed audio signal. This work focuses on using a deep learning solution to perform speaker differentiation or classification given audio signals where a single speaker is speaking at a time. The model developed in this work is presented as a first step towards solving the cocktail party problem.

The aim of this work was to evaluate how a machine learning model would perform in solving speaker differentiation in audio when the number of speaker buckets and dataset size was larger or smaller. Bucketization, which is commonly known as multivariate binning or multivariate data analysis, is the process of extracting only relevant features from given data in order to classify them [2]. In this work, relevant features were extracted from audio snippets in order to classify the snippets to belong to a specific bucket (speaker). In multidimensional data, not all features are relevant when classifying data samples [2]. As such, an approach in which only interesting features remain is necessary [2]. Traditional methods such as principal component analysis (PCA) and multidimensional scaling (MDS) have been used for relevant feature extraction given multidimensional data [2]. This work outlines the use of a machine learning solution in order to perform feature extraction as well as bucketization automatically. Many different types of machine learning models have been implemented in order to solve various problems and challenges relating to audio signals. The machine learning model used in this work was a convolutional autoencoder (CAE).

II. Background

Autoencoders (AE) are comprised of two parts, an encoder, and a decoder. The encoder will reduce the dimensions of the input to a latent space representation while the decoder will attempt to reconstruct the compressed data [3]. Downsampling is used during the encoder stage and upsampling during the decoder stage in order to compress and reconstruct the input data, respectively [4]. AEs, specifically CAEs, have been used in many applications that involve the use of image data. CAEs have been effective in extracting visual features of input images, and as a result, have allowed the generation of much more accurate latent space representations as compared to traditional AEs [4]. Figure 1 provides a visual representation of the structure of an AE.

CAEs are based on standard AEs where convolutional layers are used for encoding and/or decoding [4]. CAEs preserve

978-1-5386-6378-3/18/$31.00

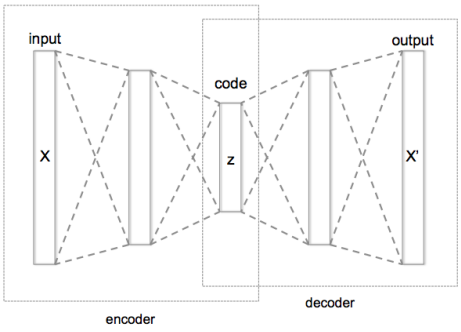

Fig. 1: Autoencoder Structure

spatial locality since the weights are shared among all input locations [4]. During training, back-propagation is used in order to compute the gradient of the error function. This allows for updating the network's weights using a gradient descent algorithm such as Adadelta [4].

III. Prior Art

Mel-frequency cepstrum (MFCC) representations have dominated the field of speech recognition in terms of feature representation as they provide a compact form of the amplitude spectrum representation of audio [5]. MFCCs provide a short-term spectral representation of audio features [5]. The work in [5] found that MFCCs did not have a negative effect on speech and music discrimination algorithms when used as a pre-processing step for feature extraction.

CAEs have been used as a solution to many different problems that range from classification to denoising. An example application of CAEs in denoising data, is in the field of medical image denoising [6]. The work in [6] outlines how the use of small sample size data along with a CAE can efficiently denoise medical images.

CAEs have also been applied to classification based problems as outlined in [7]. In [7], a CAE was used for classifying high-resolution Synthetic Aperture Radar (SAR) images that are contaminated by speckle noise [7]. The work in [7] outlines how CAEs are able to extract features as well as classify the SAR images automatically eliminating the need for manual and time-consuming feature extraction.

The field of speaker separation and classification has also made use of machine learning algorithms [8]. In [8], the aim was to solve the cocktail party problem with modern machine learning techniques and algorithms. A general regression neural network classifier was used as a solution for the cocktail party problem [8]. In [8] a feedforward multilayer artificial neural network was used to classify the attention focus of a listener in a multi-speaker setting given features extracted from electroencephalogram data. The model was evaluated based on classification accuracy, sensitivity, specificity, and computation time. The results were around 99% classification accuracy, 98.9% sensitivity, 99.1% specificity, and around 8 seconds of computational time [8].

Another example application of a machine learning solution in speaker separation is outlined in [9]. Similar to [8], in [9] a modern machine learning solution was proposed in order to solve the cocktail party problem. In [9], a solution was proposed where utterance-level permutation invariance and training was used for training a bi-directional long short-term memory recurrent neural network [9]. The resulting model was capable of improving signal-to-distortion ratio as well as the extended short-time objective intelligibility for challenging signal-to-noise ratios in multi-speaker audio [9].

IV. Developed System

In prior art (e.g. [8], [9]), non CAE machine learning solutions were implemented in order to solve speaker separation in multi-speaker audio. However, the developed systems required heavy audio pre-processing, large datasets, and proved to be computationally intensive. This work outlines the use of a modern machine learning solution in order to solve speaker differentiation in multi-speaker audio. A solution that requires minimal audio pre-processing, small sample size datasets, and was computationally efficient was designed and developed. A CAE model was trained and evaluated on audio originating from two microphones simultaneously. The aim was to evaluate the CAE's accuracy as the number of buckets at the network's output increases. The expectation was that a specific combination of dataset quality, dataset size, and bucket size would provide the best precision, recall, and accuracy by the CAE at differentiating speakers in audio. As the bucket size at the CAE's output increases – that is increasing the number of output neurons – the resulting confusion matrix will be larger. With N buckets, an N x N matrix would be constructed. The expectation was that no matter the size and quality of the dataset, as the number of buckets increase (i.e. N becomes larger), the model's accuracy would eventually decrease.

In prior art (e.g. [6], [7]), the decoder part of the CAE was used to reconstruct the data reduced to latent space representation. However, in this work, the decoder of the CAE used the relevant features in the encoded data – the data that was provided as input to the decoder – in order to differentiate speakers in the original audio by placing them within buckets. As such, the decoder in this work was simply a dense neural network (DNN) rather than a convolutional one where the flattened encoded data was provided as input.

The structure of the CAE implemented in this work is fairly straightforward. The encoder was comprised of one input layer followed by three convolutional layers, a pooling layer, and a dropout layer. The most commonly used pooling method in modern machine learning solutions, Max-Pooling, was used in order to perform dimensionality reduction [10]. Dimensionality reduction is commonly performed in the preprocessing stage of clustering and classification applications [11]. In this work, dimensionality reduction was part of the data analysis stage rather than the preprocessing stage. The reduced output of the decoder was then flattened using a flatten layer. The

flatten layer allows for the flattening of the feature maps from the decoder's output by combining them [12]. The flattened data was then provided as input to the decoder which was comprised of dense layers with dropout layers in between. Dropout layers were used as they help prevent the model from overfitting [13]. The structure of the CAE implemented in this work is outlined in more detail in Table I and Figure 2.

TABLE I: CAE For Differentiating 2 Speakers

Layer	Output Shape	Parameters
Input Layer	(40, 11, 1)	0
Conv2D Layer	(19, 10, 32)	160
Conv2D Layer	(18, 9, 48)	6192
Conv2D Layer	(17, 8, 120)	23160
MaxPooling2D Layer	(8, 4, 120)	0
Dropout Layer	(8, 4, 120)	0
Flatten Layer	(3840)	0
Dense Layer	(128)	491648
Dropout Layer	(128)	0
Dense Layer	(64)	8256
Dropout Layer	(64)	0
Dense Layer	(2)	130

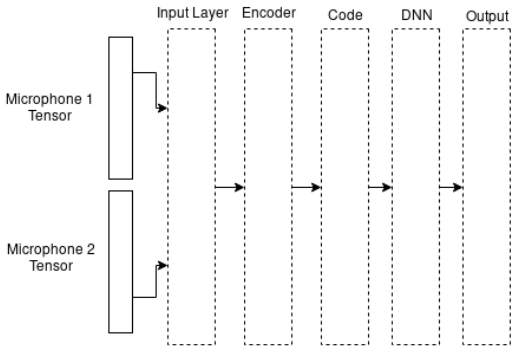

Fig. 2: Model Diagram

The activation function used for all layers, apart from the output layer, was the rectified linear unit (ReLU) as it is computationally efficient [14]. The activation function used for the output layer was softmax as the decoder was used for multi-class classification. The loss function was categorical cross-entropy with Adadelta as the optimizer. Softmax activation and the cross-entropy penalty function were chosen as there is a natural pairing between them [15]. Adadelta was used as the optimization algorithm as specifying a learning rate is unnecessary since it has been eliminated from the algorithm's update rule [16]. As such, the number of hyper-parameters was reduced as the model required less manual tuning of the learning rate.

Prior to training and testing the model, the audio datasets were preprocessed as follows. The work in [5], demonstrated that MFCCs can provide a short-term spectral representation of audio features without negatively affecting speech and music discrimination algorithms. As such, in order to improve the computational efficiency of the system developed in this work without a reduction in accuracy, the input data was preprocessed by transforming each audio snippet into MFCC representation with a window length of 2048 and hop length of 512. In order to compute the MFCCs of the audio snippets used in this work, the python package Librosa was used. Librosa is a python package that is commonly used in the field of music in order to retrieve information from audio data [17]. Converting the audio data into MFCC representation resulted in image-like 2D data that was similar to data from [6] [7]. For each audio file, the preprocessing resulted in a 20 x 11 tensor. In this case, the Librosa MFCC function computed 20 MFCCs over 11 frames. Each pair of identical audio snippet's MFCC tensor representations, one from each microphone, were concatenated in order to serve as input to the CAE developed in this work. As such, the shape of the input data for the model was $(40, 11)$.

V. Data Preparation

Training and testing data were collected using two microphones simultaneously. Audio snippets were collected from a total of 6 speakers under the age of 30, 3 of which were male and 3 of which were female. The speakers were also a mix of both native and non-native English speakers. During recording, the speakers were placed at a distance of 2m away from both microphones, with the two microphones at a distance of 1m apart. Each microphone was attached to a Raspberry Pi 3 Model V1.2 running on Rasbian OS where the recordings were started, stopped, and saved. This allowed for identical separate audio snippets from each microphone.

The speakers were given the same transcripts from Mozilla's Common Voice dataset. Each speaker read the same list of sentences, where each sentence was recorded by each microphone and saved to a WAV file. Each pair of WAV files generated were considered as a single sample in the dataset used in this work. Each WAV file had a sampling rate of 44100Hz and a duration of 10 seconds with the noise level in the room at an average of 47dB during recordings. A total of 82 recordings for each speaker from each microphone were collected for the training dataset, and a total of 12 recordings for each speaker from each microphone were collected for the testing dataset. Both datasets were organized and labeled according to the speaker present in the audio snippet and the source microphone used to record that snippet. Each speaker would represent a bucket at the model's output.

VI. Results

The model was trained with a batch size of 64 over 80 epochs – that is 80 full passes of the training dataset through the CAE – using a dataset of 82 audio snippets per speaker. The training dataset was split between a training subset of 70 audio snippets per speaker and a subset of 10 audio snippets per speaker for evaluating the model. The model's accuracy

was evaluated by constructing confusion matrices where the true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) values were computed. Precision and recall were then calculated using TP, FN, and FP. Table II provides the recall, precision, and accuracy for each confusion matrix.

TABLE II: Confusion Matrix Results

# of Buckets	Recall (%)	Precision (%)	Accuracy (%)
2	100	100	100
3	89.01	100	94.51
4	97.43	97.22	97.33
5	90.38	91.67	91.03
6	72.11	76.39	76.57

Table II clearly demonstrates that as the number of buckets increase, the model's accuracy decreases. The following diagram provides a visualization of the model's decline in accuracy as the number of buckets increase.

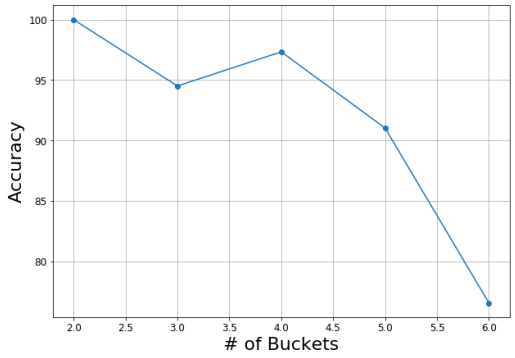

Fig. 3: Model Accuracy vs. Number of Speakers Classes

The results extracted from the generated confusion matrices indicate that a CAE can differentiate speakers with audio originating from two microphones simultaneously. The results also indicate that the larger the number of buckets the model was trained on, the less accurate the model was. As such, the hypothesis that as the number of buckets increase, the model's accuracy would eventually decrease has been confirmed. However, the limited number of testing samples must be taken into account. Data quality and quantity have a direct impact on the performance of machine learning algorithms [18]. As such, in order to verify the results extracted from the confusion matrices, k-fold cross-validation was also used to evaluate the model. In this work, the model was evaluated using 10 folds where the model's performance was evaluated on the held-out validation fold [19]. Table III and Figure 4 provide the k-fold cross-validation results.

Table III and Figure 4 further demonstrate that a CAE can differentiate speakers in audio. However, as previously noted based on the results from the confusion matrices, Table III and Figure 4 also demonstrate that as the number of speakers

TABLE III: K-Fold Cross Validation Results – 2 Microphones

# of Buckets	Mean Accuracy (%)	Standard Deviation (%)
2	99.29	2.14
3	97.62	3.19
4	96.43	3.19
5	93.43	3.39
6	88.10	2.82

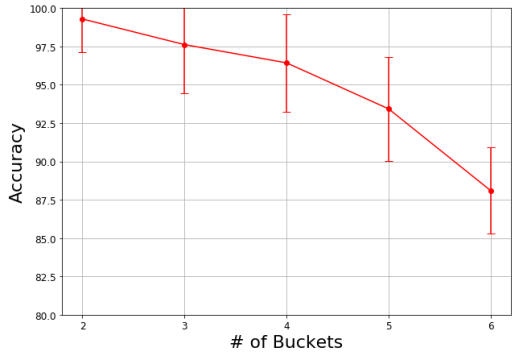

Fig. 4: KFold Cross Validation Results – 2 Microphones

increase, the model's accuracy decreases. The results obtained using k-fold cross validation also appear to be slightly better than those obtained with confusion matrices. In k-fold cross validation, the training dataset is randomly divided into k subsets, or folds, where the model trains on each subset except for one. The model's performance is evaluated on the held out validation subset [19]. In this work, the model was tested on 10 folds from a dataset of 82 samples. The model was trained 10 times and evaluated each time on the random held out subset or fold. The model's performance was then evaluated by taking the average of the accuracy results from each held out evaluation fold. As such, given the limited dataset used in this work, k-fold cross validation provides lower variance and reduces bias in the testing subset as compared to a single hold-out set when evaluating the model.

In data preprocessing, the resulting tensors from each microphone were concatenated and provided as input to the model. Changing the input shape of the model from $(40, 11)$ to $(20, 11)$ and providing the resulting tensor from just one microphone, allowed for evaluating the model's performance when using a single microphone. Table IV and Figure 5 provide the k-fold cross-validation results when using a single microphone and clearly demonstrate that the model performed better when given audio from two microphones.

VII. Discussion and Conclusion

The findings in this work clearly demonstrate that a CAE can accurately be used in differentiating speakers given a small sample size of audio collected from two microphones simultaneously. With cross-validation, the model was able to differentiate between two, three, four, five and six speaker

TABLE IV: K-Fold Cross Validation Results – 1 Microphone

# of Buckets	Mean Accuracy (%)	Standard Deviation (%)
2	94.44	3.36
3	95.19	2.94
4	93.30	6.80
5	84.86	4.08
6	82.27	8.07

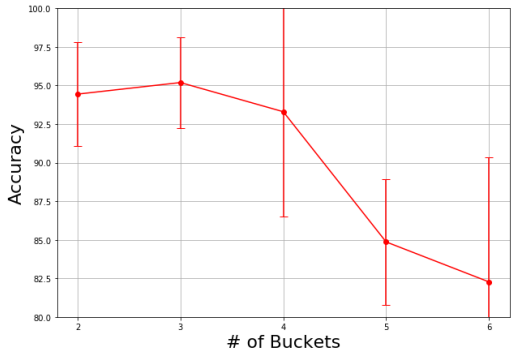

Fig. 5: KFold Cross Validation Results – 1 Microphone

classes with an accuracy of about 99%, 97%, 96%, 93%, and 88%, respectively. The results also indicate that as the number of speaker classes increases, the model's accuracy decreases. This confirms the hypothesis outlined at the start of this work that no matter the size and quality of the dataset, as the number of buckets increases, the model's accuracy would eventually decrease. Nevertheless, given the accuracy results, when given up to 5 different speakers, the model was still able to differentiate between speakers in audio collected from two microphones simultaneously with an accuracy of over 90%. Results also demonstrate that the model performed better when given audio from two microphones simultaneously as compared to a single microphone. Using cross-validation and with preprocessed audio data collected from a single microphone, the model was able to differentiate between two, three, four, five, and six speaker classes with an accuracy of about 94%, 95%, 93%, 85%, and 82%, respectively. As such, with the use of two microphones simultaneously to collect speaker audio, results demonstrate that using a CAE in solving multi-speaker classification in audio is both an efficient solution as well as a performant one.

Further investigation into the drop in the model's accuracy as the number of buckets increase could include using larger sample size datasets as the number of buckets increase. It is possible that an increase in the number of training samples when given a larger number buckets may eliminate the drop in the model's accuracy. Other attempts at improving the model's performance may include increasing the quality of the training dataset or using other sound pre-processing techniques other than MFCC. MFCC eliminates a large amount of information from the original audio wave, using other techniques that conserve more information such as Short-Term Fourier Transform (STFT) may allow the model to more accurately differentiate between speakers in audio.

VIII. Acknowledgments

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) and of Unified Computer Intelligence Corporation, Canada.

References

[1] Y. Isik, J. L. Roux, Z. Chen, S. Watanabe, and J. R. Hershey, "Single-channel multi-speaker separation using deep clustering," *arXiv preprint arXiv:1607.02173*, 2016.

[2] K. Lu, H.-W. Shen *et al.*, "Multivariate volumetric data analysis and visualization through bottom-up subspace exploration," in *2017 IEEE Pacific Visualization Symposium (PacificVis)*. IEEE, 2017, pp. 141–150.

[3] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of machine learning research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[4] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *International Conference on Artificial Neural Networks*. Springer, 2011, pp. 52–59.

[5] B. Logan *et al.*, "Mel frequency cepstral coefficients for music modeling." in *ISMIR*, vol. 270, 2000, pp. 1–11.

[6] L. Gondara, "Medical image denoising using convolutional denoising autoencoders," in *Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on*. IEEE, 2016, pp. 241–246.

[7] J. Geng, J. Fan, H. Wang, X. Ma, B. Li, and F. Chen, "High-resolution sar image classification via deep convolutional autoencoders," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 11, pp. 2351–2355, 2015.

[8] P. Shree, P. Swami, V. Suresh, and T. K. Gandhi, "A novel technique for identifying attentional selection in a dichotic environment," in *India Conference (INDICON), 2016 IEEE Annual*. IEEE, 2016, pp. 1–5.

[9] M. Kolbæk, D. Yu, Z.-H. Tan, and J. Jensen, "Joint separation and denoising of noisy multi-talker speech using recurrent neural networks and permutation invariant training," in *Machine Learning for Signal Processing (MLSP), 2017 IEEE 27th International Workshop on*. IEEE, 2017, pp. 1–6.

[10] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *Engineering and Technology (ICET), 2017 International Conference on*. IEEE, 2017, pp. 1–6.

[11] B. Yang, X. Fu, and N. D. Sidiropoulos, "Learning from hidden traits: Joint factor analysis and latent clustering," *IEEE Transactions on Signal Processing*, vol. 65, no. 1, pp. 256–269, 2017.

[12] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," in *International Conference on Web-Age Information Management*. Springer, 2014, pp. 298–310.

[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[14] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[15] R. A. Dunne and N. A. Campbell, "On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function," in *Proc. 8th Aust. Conf. on the Neural Networks, Melbourne*, vol. 181. Citeseer, 1997, p. 185.

[16] S. Ruder, "An overview of gradient descent optimization algorithms, 2016," *arXiv preprint arXiv:1609.04747*, 2016.

[17] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, 2015, pp. 18–25.

[18] V. Sessions and M. Valtorta, "The effects of data quality on machine learning algorithms." *ICIQ*, vol. 6, pp. 485–498, 2006.

[19] T. Wong and N. Yang, "Dependency analysis of accuracy estimates in k-fold cross validation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 11, pp. 2417–2427, Nov 2017.