Contrastive Predictive Coding based Unupervised Pre-Training for Speaker Classification using MFCC Spectrograms

Dan Lehman - 1495739, Twan Leloup - 1009272
Advanced Sensing using Deep Learning - Group 18
Artificial Intelligence & Engineering Systems, Eindhoven University of Technology
d.lehman@student.tue.nl, t.r.leloup@student.tue.nl

Abstract

This work concerns the implementation of Mel-Frequency Cepstral Coefficient (MFCC) spectrograms for training a speaker identification network in a fully supervised manner and in an unsupervised manner, using Contrastive Predictive Coding (CPC). We find that our models, trained with on spectrograms, did not surpass the performance of the baselines implemented by [9], which were trained on raw audio. Additionally, we conducted a thorough analysis of our models' performance when using different amounts of labeled data during training. Most notably, the classification model, whose encoder was pretrained with CPC, outperformed the randomly initialized network when trained with only 5% and 10% of the labels.

1. Introduction

Developments in AI have taken the world by storm in recent years. A major contributor to its rising popularity has been the stunning achievements reached by Deep Learning, be it through photo-realistic image generators or human-like natural language models. One of the remaining challenges of training deep neural networks is the availability of annotated data with which the networks can be trained [6]. To train in a fully supervised manner, a lot of high quality annotated data is required, which can be very costly, as it often requires humans to hand-annotate entire datasets.

An approach to deep learning with the potential to tackle this challenge is known as Contrastive Predictive Coding (CPC) [9]. CPC allows models to be pretrained in an unsupervised manner, meaning that it can be leveraged to train an encoder to extract features without using any annotated labels. Thus, when pretraining an encoder with CPC, less hand-annotated labels are needed to reach the equivalent performance of a network trained in a fully supervised manner [3]. A detailed explanation of the workings of CPC is provided in section 3.

Deep neural networks are known to perform well at speaker identification and the systematic review by [8] asserts that most papers use some form of Mel-Frequency Cepstral Coefficients (MFCC) feature extraction for speaker identification, regardless of the classification algorithm used. Our work builds on the work of [9], as we test the use of CPC on text-independent speaker identification, meaning that the our network needs to be able to identify a speaker regardless of the words being spoken. However, our methods rely on the use of MFCC spectrograms as an input to the network instead of raw audio. This preprocessing of the audio into MFCC spectrograms transforms the input to a more useful representation for speaker identification, as the spectrograms depict the frequencies present in audio signals on the Mel scale, which models the non-linear human sensitivity to different frequencies of sound [4]. To the best of our knowledge, CPC has not been tested on speaker identification using MFCC spectrograms. We compare the performance of an encoder pretrained using CPC to the same encoder trained in a fully supervised manner, on the same dataset. To measure the effects of using MFCC spectrograms, both trained models are compared to the baseline provided by [9]. Additionally, this paper aims to assess whether contrastive predictive coding can be leveraged to reduce the amount of labeled data required to train a speaker identification network without suffering a loss in accuracy.

2. Related Work

Contrastive predictive coding represented a breakthrough in unsupervised learning. In 2018, the authors of [9] introduced CPC as a means for unsupervised learning on a number of tasks, including image classification, speaker identification and reinforcement learning. They showed that a speaker identification network trained with CPC achieved an accuracy of 97.4% compared to 98.5% when using fully supervised learning on a subset of the LibriSpeech dataset with 251 speakers.

More recent work evaluated the CPC framework for image classification on the PASCAL VOC dataset by varying the amount of labeled data used [3]. The researchers concluded that by using CPC, they could achieve equivalent accuracies to supervised learning with 2-5 times less labeled

data. These results show the advantage of CPC when limited amounts of labeled data are available for an image classification problem. Our work aims to conduct the same experiment on the task of speaker identification.

3. Methodology

In this section, we first provide an overview of how CPC works, how we use MFCC spectrograms, and how both of them have been implemented in our work.

CPC works by training an encoder g_{enc} to map crops of data to a lower-dimensional latent space z and tries to predict the latent embeddings of one or more crops using the features extracted in preceding crops. More precisely, g_{enc} generates the latent encodings z_{t-n_p} to z_t corresponding to the consecutive crops x_{t-n_p} to x_t, where n_p is the number of 'past' crops considered. The latent encodings are then passed through an autoregressive model $g_{ar}(z_{t-n_p}, ..., z_t)$, which outputs the context vector c_t, as can be seen in Figure 1. Future latent encodings are predicted using a linear transformation of the aforementioned context vector c_t, giving rise to distinct predicted latent embeddings $\hat{z}_{t+1}$ to $\hat{z}_{t+n_f}$, where n_f is the number of future latent embedding to predict. Simultaneously, the true latent embeddings z_{t+1} to z_{t+n_f} are generated by the same encoder g_{enc}. The complete CPC structure can be found in Figure 1.

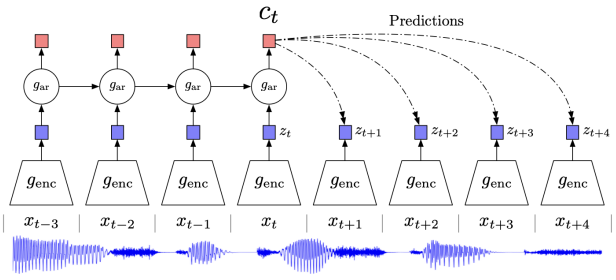

Figure 1. Visualisation of the CPC framework (image retrieved from [9])

The predicted and true latent embeddings are needed in order to devise a loss function. The so-called Info Noise-Contrastive Estimation (InfoNCE) loss proposed by [9] was implemented. It forces a model to extract latent features in such a way that the mutual information between present and future latent embeddings is maximized [3, 9]. To model the mutual information between a sample's future latent embedding and context vector, a density ratio f is defined for the k-th latent encoding as:

$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right), \quad (1)$$

where W_k represents the linear transformation depicted by the dotted arrows in Figure 1. The InfoNCE loss is set up such that the density ratio f_k is maximized when computed between the context vector and a 'positive sample', compared to other 'negative samples'. Here, positive sample refers to the true future latent embedding z_{t+k} inferred by g_{enc}. On the other hand, negative samples refer to the latent embeddings of crops of other randomly chosen audio files in the same batch that (likely) correspond to different speakers and are therefore uncorrelated with the true latent encoding z_{t+k}.

Given that MFCC spectrograms are commonly used to enhance deep learning based speaker identification [8], we have implemented them as a preprocessing step to our neural network. Mel Frequency Cepstrum Coefficients constitute a number of coefficients that are used as features. Cepstra are obtained by taking the Inverse Fourier Transform of the logarithm of the signal spectrum [2]. They are then converted to the Mel scale, which models the non-linear human perception of sound, thus only retaining information that is most relevant for human speech [1].

To measure the classification performance of a model pre-trained with CPC, we pre-trained one model with CPC and trained another model in a fully supervised manner, for comparison. They both used the same training data, preprocessed as MFCC spectrograms, and they all have exactly the same model architecture: a convolutional encoder with one fully connected layer for classification. The CPC-pretrained model was evaluated in two different ways. First, by training the classifying layer once with locked encoder weights. We name this model CLE (CPC-pretrained model with Locked Encoder). Secondly, we train the same model, allowing the encoder weights to be finetuned, which we call CUE (CPC-pretrained model with Unlocked Encoder). The model trained without unsupervised pretraining is called FS (Fully Supervised). The following subsection describes the implementation details of our experiments so that they can be reproduced.

3.1. Implementation details

Our MFCC spectrograms have a height of 64 and a width of 96 pixels. This represents the 64 frequency bins and 96 time bins that each audio sample is represented as. The latter were obtained by using a window length of 430 time samples for computing the Fast Fourier Transforms. Each spectrogram corresponds to an audio crop of length 1.28s, the same length of audio used by [9]. All models tested in this paper were trained with the same crop size and MFCC hyperparameters.

The encoder architecture is comprised of a 2D convolutional neural network. It consists of six convolutional layers, each with a relu activation function, (2,2) max pooling, batch normalization and Kaiming normal weight initialization. The convolutions all have a stride of 1 and zero-padding to preserve spatial dimensions according to their respective kernel sizes (5,3,3,3,3,3). The six max pooling

layers lead to a downsampling factor of 64. To make our results comparable to the work in [9], we tried to replicate their implementation details as closely as possible. Therefore, the dimensionality of our latent vectors z is 512. Thus, when training our model with the classifying head, this last layer of the network consists of a fully connected layer with softmax activations, with 512 inputs and 251 outputs, one for each speaker in the dataset. The autoregressive model g_{ar} used when pretraining our model with CPC, consists of a single layer GRU with input size 512 and hidden layer size 256. Thus, the context vector used to predict future latent embeddings is of size 256 and the predictions (linear transformations W_k) consist of a fully connected layer with 256 and 512 in- and output dimensions, respectively.

The Adam optimizer was used for all our models, along with one-cycle learning rate scheduling, for which a minimal and maximal learning rate were specified [7] according to the hyperparameter selection procedure outlined in subsection 4.2.

4. Experiments

This section concerns the details of the conducted experiments, the dataset used, and the method used to find optimal hyperparameters. Firstly, to assess whether CPC can help a model's performance when less labeled data is available, the FS, CLE and CUE models were trained on 1%, 2%, 5%, 10%, 20%, 50% and 100% of the labeled data. In all cases, CLE and CUE were initialized with the encoder weights trained by CPC on 100% of the available data without labels. The performances of the different models are compared using their classification accuracy.

4.1. Dataset

To make our results comparable to the work from [9], we used the same subset of the LibriSpeech dataset [5]. It contains 100 hours of (speaker, phone and text) labeled audio data at a sampling rate of 16kHz, split into 80 hours of training and 20 hours of validation data. The dataset contains 251 different speakers, and there are a total of 22830 audio files in the training set, ranging between 1.4 and 24.5 seconds in length. However, for the experiments performed in this paper, a part of the training data had to be discarded as samples that were too short could not be used when training with CPC, as it requires samples to be sufficiently long to generate latent embeddings for $n_p + n_f + 1$ consecutive windows of audio. Since we used a window size of 1.28s, with one past sample ($n_p = 1$) and five future predictions ($n_f = 5$), this meant that all training samples shorter than 8.96s in length were discarded. This represents 16% of the number of training samples, but only 7% of the length of the training set in terms of time. The choices for n_p and n_f are motivated in subsection 4.2.

Table 1. Hyperparameters for different experiments. P-CPC = Pretraining with CPC

	FS	P-CPC	CLE	CUE
Batch size	26	33	64	61
Min. learning rate	3.7e-3	1e-3	2e-4	3.1e-3
Max. learning rate	5.7e-3	5e-3	6.8e-4	2.9e-2

4.2. Hyperparameter selection

To select optimal hyperparameters for each of our experiments, a random search algorithm was implemented. For FS, this consisted of training the model for 20 epochs 30 times, each time randomly varying the batch size between 10 and 64, the minimum learning rate between 1e-5 and 1e-2, and the maximum learning rate between 1 and 10 times the minimum learning rate. Those hyperparameters above were randomly chosen in the same way for training with CPC, but with the addition of varying the number of future latent predictions n_f between 1 and 10, the number of negative samples between 2 and the batchsize, and the number of past latent embeddings n_p between 1 and $12 - n_f$. For the FS network, the optimal hyperparameters were chosen based on the best validation score after 20 epochs of training. For the CPC-pretrained encoder, the optimal hyperparameters were chosen based on the validation score of a classifier trained for an additional 10 epochs with frozen CPC-trained encoder weights. Then, the weights of the 'best' encoder were used to perform a random search to find the optimal hyperparameters for CLE and CUE. Table 1 contains the optimal hyperparameters found, which were used for each of our experiments. Each of our models was trained until convergence of the validation loss. Additionally, the optimal CPC hyperparameters were found to be $n_f = 5$, $n_p = 1$ and the number of negative samples equal to 16.

5. Results and discussion

The results of our experiments are reported in Table 3 and visualized in Figure 2. It can be observed that CUE performed best by a significant margin only when using 5% or 10% of the labels. When using more labels, FS performed best (by a small margin) and CLE performed significantly worse. By comparison, when using fewer labels (1 to 2%), all three models performed more or less similarly.

Table 2 compares our results to those of [9], showing that our FS model performed slightly worse than theirs, and more notably, our CLE model performed significantly worse than theirs. Thus, our results do not show that using MFCC spectrograms leads to improvements in accuracy when training a network for speaker classification, whether it be trained in a fully supervised manner or with CPC.

However, it cannot be concluded that training a fully su-

pervised or CPC-pretrained network with MFCC spectrograms compared to raw audio produces worse results. This is because our implementation does not precisely copy that of [9], because the authors did not provide sufficient details for their results to be reproduced. For instance, they did not indicate their number of future and past latents used when training with CPC, nor whether training data was omitted due to insufficient audio length, as described in subsection 4.1. Thus, the MFCC preprocessing is not the only variable that changed between our experiments and those of [9]. For a fairer comparison, we would have to retrain our own models using raw audio as an input.

Similarly, the CPC-pretrained model in [9] appears to have been trained by attaching a fully connected layer to the context vector c_t instead of the latent embedding z_t as for our model. This means that their CPC-pretrained network could benefit from past latent embeddings and additional nonlinear layers from the autoregressive model. However, this does not make for a fair comparison to a model trained in a fully supervised manner.

It makes a lot of sense that our CUE model would not exhibit worse performance than the FS model as the CUE model is being trained in a fully supervised manner, with weights initialized according to the CPC-pretrained encoder. Nevertheless the pretraining makes the CUE perform significantly better, especially when using 10% of the labels, meaning that the effect of the pretrained weight initialization is not lost when subsequently training the model in a supervised fashion. Still, ideally, the feature extraction abilities of CPC should be measured by freezing the encoder weights. One potential reason why our CLE model performed poorly is that, even if the encoder was able to extract useful features, a single fully connected layer might not be sufficient to capture the nonlinear relationship between the extracted features and the 251 speaker classes.

Table 2. Accuracy of our models (using MFCC) and those of [9] (trained on raw audio), using all labels

	Work of [9]	**Ours**
CLE	97.4	42.3
CUE	N/A	97.8
FS	98.5	96.9
Percentage of training data used	100*	84

* it is not specified whether [9] discarded training samples

Table 3. Accuracy [%] of models on different amounts of labels.

	Percentage of labels used						
Model	1	2	5	10	20	50	100
FS	6.76	11.1	15.6	23.5	76.6	93.9	97.8
CLE	3.93	7.68	14.3	15.8	24.3	28.9	42.3
CUE	5.48	8.62	23.6	45.7	72.7	93.3	96.9

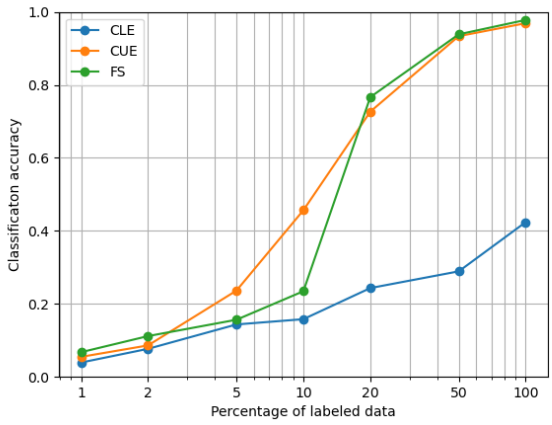

Figure 2. Comparison of three speaker identification methods on MFCC spectrograms.

5.1. Future work

As mentioned previously, our work could be extended by reproducing our experiments using raw audio as a input. It might also be interesting to train a fully connected layer on the MFCC spectrograms to compare the feature extraction of the CLE model to the quality of the features at the input of the network. Additionally, an ablation study could be run to investigate the effects of varying the number of past and future latent encodings when training CPC. Lastly, more broadly, our experiments could be extended to include running inference on a distinct dataset (perhaps containing different languages) using both our FS and CLE model. Since the classes (speakers) would be different, for both models we would be required to freeze the encoder and only finetune the fully connected layers on the new class. Such an experiment could be used to exploit and quantify a distinct benefit of CPC: its generalizability to unseen classes.

6. Conclusion

To conclude, we have shown that, when fewer training labels are available, pretraining an encoder using CPC on the entire dataset can lead to significantly better classification under the right circumstances, i.e. when 5 or 10% of labels are available and training with unlocked encoder weights. However, we have also shown that the features extracted by our models with MFCC spectrograms instead of raw audio as input when pretrained with CPC are not powerful enough to make high accuracy classifications when freezing the encoder and training a classifier on top of it. Additionally, we have discussed potential reasons why this might be the case and why our results are not directly comparable to the baseline from the experiments of [9] as well as a number of ways that our work could be extended.

References

[1] J.P. Campbell. Speaker recognition: a tutorial. *Proceedings of the IEEE*, 85(9):1437–1462, 1997. 2

[2] Herbert Gish and Michael Schmidt. Text-independent speaker identification. *IEEE signal processing magazine*, 11(4):18–32, 1994. 2

[3] Olivier J. Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, S. M. Ali Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding, 2020. 1, 2

[4] Fang-Yie Leu and Guan-Liang Lin. An mfcc-based speaker identification system. In *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, pages 1055–1062, 2017. 1

[5] Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert. MLS: A large-scale multilingual dataset for speech research. In *Interspeech 2020*. ISCA, oct 2020. 3

[6] Christoffer Bøgelund Rasmussen, Kristian Kirk, and Thomas B. Moeslund. The challenge of data annotation in deep learningmdash;a case study on whole plant corn silage. *Sensors*, 22(4), 2022. 1

[7] Leslie N. Smith. No more pesky learning rate guessing games. *CoRR*, abs/1506.01186, 2015. 3

[8] Sreenivas Sremath Tirumala, Seyed Reza Shahamiri, Abhimanyu Singh Garhwal, and Ruili Wang. Speaker identification features extraction methods: A systematic review. *Expert Systems with Applications*, 90:250–271, 2017. 1, 2

[9] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2019. 1, 2, 3, 4