Effects of Layer Freezing when Transferring DeepSpeech from English to German

Onno Eberhard[1] and **Torsten Zesch**

Language Technology Lab
University of Duisburg-Essen
[1]onno.eberhard@stud.uni-due.de

Abstract

In this paper, we train Mozilla's DeepSpeech architecture in various ways on Mozilla's Common Voice German language speech dataset and compare the results. We build on previous efforts by Agarwal and Zesch (2019) and reproduce their results by training the model from scratch. We improve upon these results by using an English pretrained version of DeepSpeech for weight initialization and experiment with the effects of freezing different layers during training.

1 Introduction

The field of automatic speech recognition is dominated by research specific to English. There exist plenty available text-to-speech models pretrained on (and optimized for) the English language. When it comes to German, the range of available pretrained models becomes much sparser. In this paper, we train Mozilla's implementation[1] of Baidu's DeepSpeech architecture (Hannun et al. 2014) on German speech data. We use transfer learning to leverage the availability of a pretrained English version of DeepSpeech and observe the difference made by freezing different layers during training. The rationale for using transfer learning is not only that English and German are closely related languages. In fact, one could argue that they are very different in this context, because DeepSpeech is trained to directly infer written characters from audio data and English and German pronunciations of some characters differ greatly. However, the first few layers of the DeepSpeech network are likely not infering the final output character, but rather lower lever features of the spoken input, such as phonemes, which are shared across different languages. Thus this approach should also work for languages which are not related at all. It is to be expected that the model should give better results when trained on a small dataset than a model trained from scratch, because it does not have to learn these lower level features again.

[1]https://github.com/mozilla/DeepSpeech

2 Training

2.1 DeepSpeech architecture

Mozilla's DeepSpeech implementation differs in many ways from the original model presented in (Hannun et al. 2014). The architecture is described in detail in the official documentation[2] and is depicted in Figure 1. From the raw speech data, Mel-Frequency Cepstral Coefficients (Imai 1983) are extracted and passed to a 6-layer deep recurrent neural network. The first three layers are fully connected with a ReLU activation function. The fourth layer is a Long Short-Term Memory unit (Hochreiter and Schmidhuber 1997). The fifth layer is again fully connected and ReLU activated. The last layer outputs probabilites for each character in the language's alphabet. It is fully connected and uses a softmax activation for normalization. The character-probabilites are used to calculate a Connectionist Temporal Classification (CTC) loss function (Graves et al. 2006). The weights of the model are optimized using the Adam method (Kingma and Ba 2014) with respect to the CTC loss.

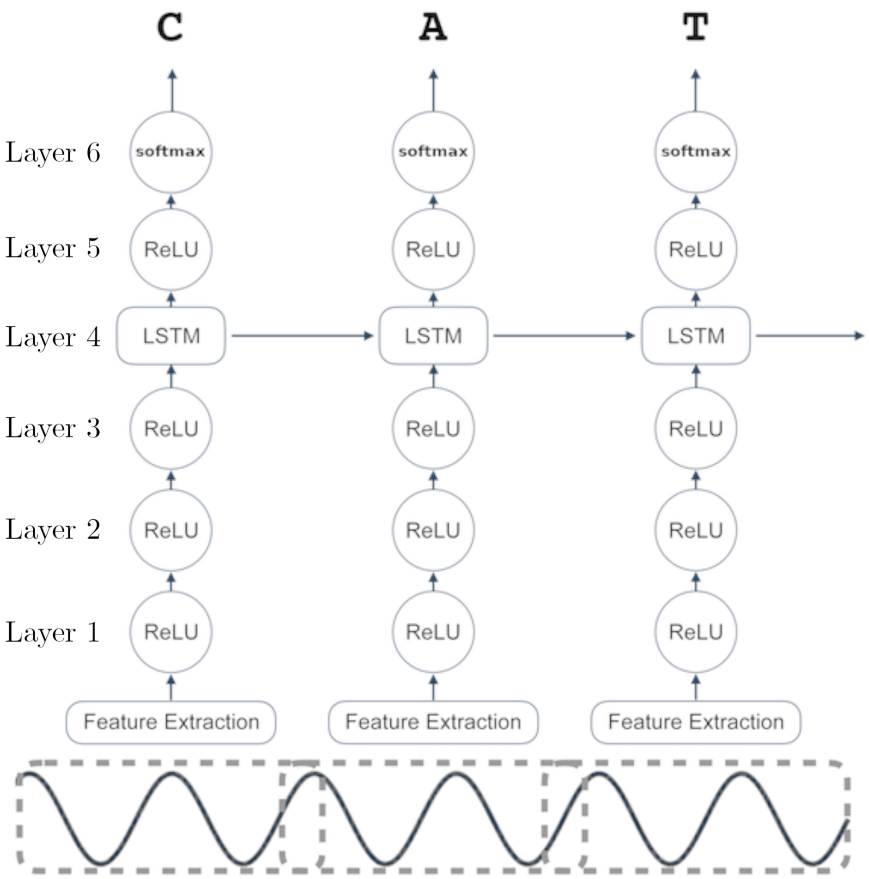

Figure 1: DeepSpeech architecture (adapted from the official documentation[2])

[2] https://deepspeech.readthedocs.io/en/latest/DeepSpeech.html

2.2 Training Details

In transfer learning, all weights of the model are initialized to those of the Enlish pretrained model, which is provided by Mozilla[3]. In addition to transfer learning, we also train the model from scratch with random weight initialization, thereby reproducing a result from Agarwal and Zesch (2019). In total, we train 6 different models:

1. The whole model from scratch ("Complete Training")
2. The model with weights initialized to those of the English pretrained model ("Simple Transfer")
3. The English-initialized model with the first layer frozen
4. The English-initialized model with the first two layers frozen
5. The English-initialized model with the first three layers frozen
6. The English-initialized model with the first three and the fifth layer frozen

We used Mozilla's DeepSpeech version 0.7.4 for training the model from scratch and version 0.7.3 for the transfer learning approach. The complete training script is available online[4]. The modified versions of DeepSpeech that utilise layer freezing are also available online[5]. The weights were frozen by adding `trainable=False` at the appropriate places in the TensorFlow code. For all models, we had to reinitialize the last layer, because of the different alphabet sizes of German and English.

We trained the models on the German-language Mozilla Common Voice 2.0 speech dataset[6]. For inference and testing we used the language model KenLM (Heafield 2011), trained on the corpus described in (Radeck-Arneth et al. 2015, Section 3.2). The text corpus consists of a mixture of text from the sources Wikipedia, Europarl and crawled sentences. The whole corpus was preprocessed with MaryTTS (Schröder and Trouvain 2003) and cleaned in the same way as described in (Agarwal and Zesch 2019). All these steps were chosen to be identical to (Agarwal and Zesch 2019), such that comparison of results would be as meaningful as possible.

In training each model, we used a batch size of 24, a learning rate of 0.0005 and a dropout rate of 0.4. We did not perform any hyperparamter optimization. These hyperparameters, as well as those selected for traning the language model, differ from those chosen in (Agarwal and Zesch 2019). This explains why our results differ even though the same model and training data was used. The training was done on a Linux machine with 96 Intel Xeon Platinum 8160 CPUs @ 2.10GHz, 256GB of memory and an NVIDIA GeForce GTX 1080 Ti GPU with 11GB of memory. Training each for 30 epochs took approximately one hour each.

[3] `https://github.com/mozilla/DeepSpeech/releases`

[4] `https://github.com/onnoeberhard/deepspeech-paper/blob/master/training.sh`

[5] `https://github.com/onnoeberhard/deepspeech-transfer`, the different versions with a different number of frozen layers are in the branches *transfer-1*, *transfer-2*, *transfer* and *transfer-4*.

[6] `https://voice.mozilla.org/en/datasets`

3 Results

The test results from the six different models described in section 2.2 are compiled in table 1. For testing, the epoch with the best validation loss during training was taken for each model. The last row is taken from (Agarwal and Zesch 2019, Table 3) and describes only the result of training with the same data as we did. Agarwal and Zesch (2019) achieved much better results when training on larger datasets. The difference between our result without transfer learning (WER 0.697) and theirs (0.797) likely stems from the difference in hyperparameters used. Figures 2 and 3 show the learning curves for all training procedures, with the green line (3 frozen layers) being the same in both plots. The epochs used for testing are also marked in the figures.

Method	WER
Complete Training	0.697
Simple Transfer	0.627
1 Frozen Layer	0.483
2 Frozen Layers	0.443
3 Frozen Layers	**0.437**
4 Frozen Layers	0.462
Agarwal and Zesch 2019	0.797

Table 1: Testing results (Word Error Rate)

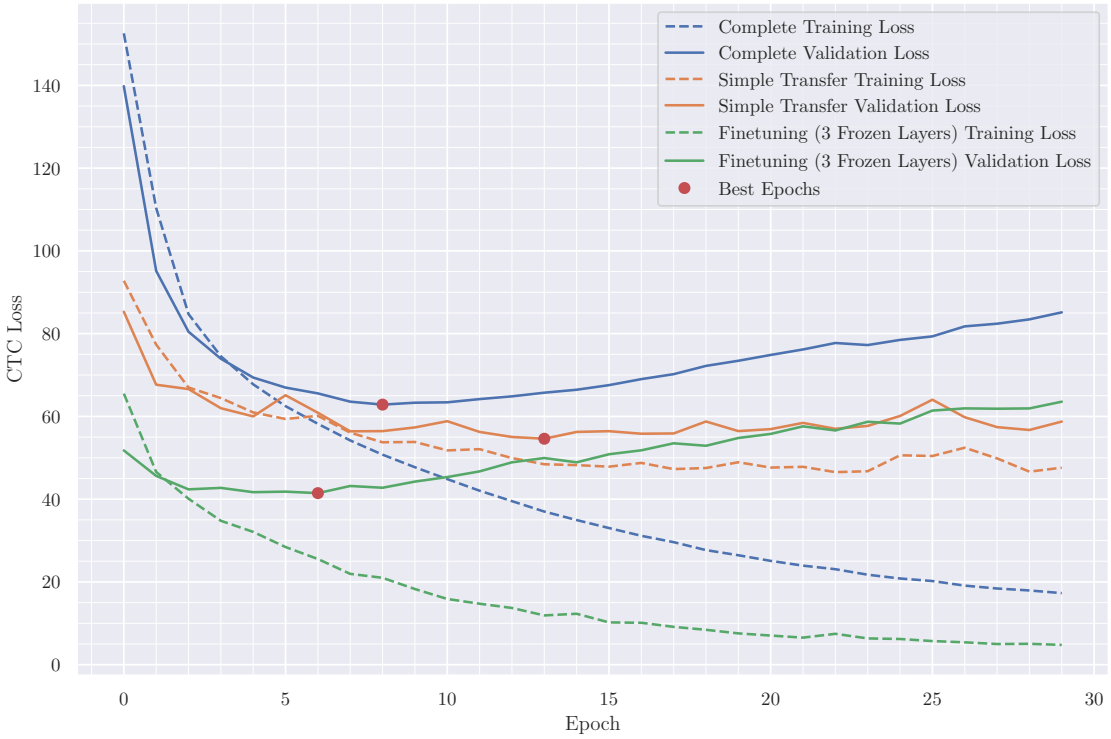

Figure 2: Learning curves: With and without transfer learning and layer freezing

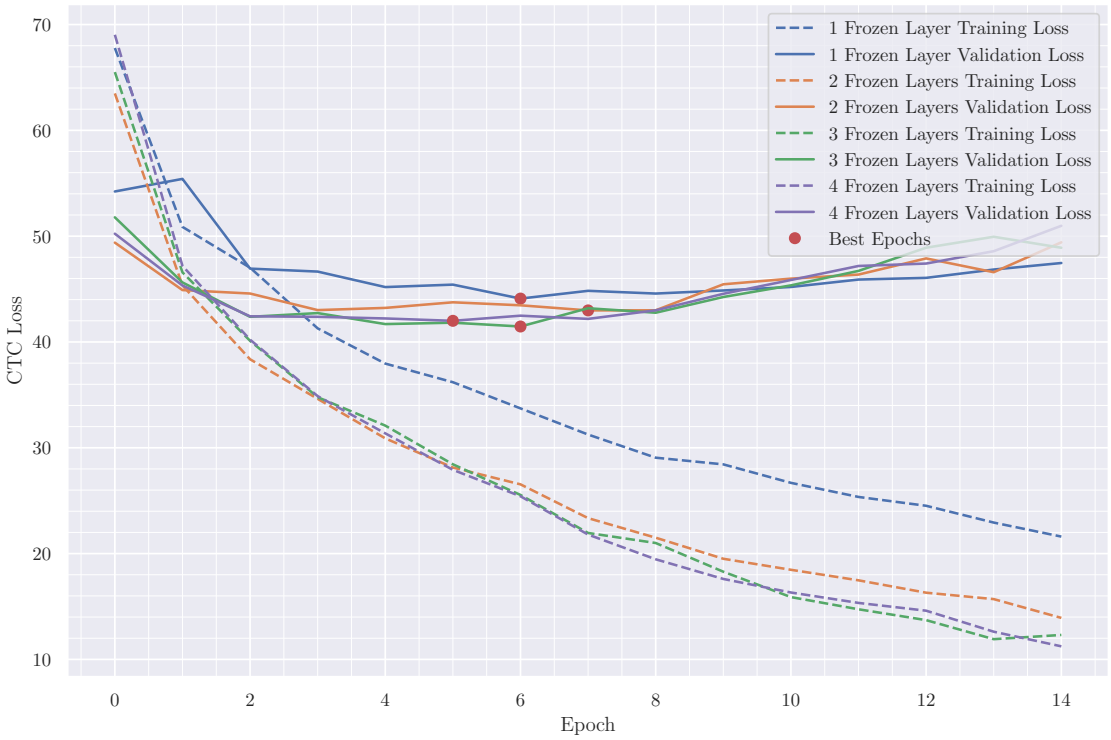

Figure 3: Learning curves: Comparison of freezing a different number of layers

The best results were achieved by the model with the first three layers frozen during training. It is notable however, that the other three models that utilise layer freezing are not far off. The training curves look remarkably similar (see Figure 3). All four models achieve much better results than the two models without layer freezing. The results seem to indicate that freezing the first layer brings the largest advanage in training, with diminishing returns on freezing the second and third layers. Additionally freezing the fifth layer slightly worsens the result.

The model with four frozen layer could only optimize the LSTM layer and the very last layer. It is surprising that it still achieves good results. It might be interesting to see what happens when the LSTM layer is frozen. It is probable that with a larger dataset the benefits of freezing weights decrease and better results are achieved with freezing fewer or no layers. However, for languages or dialects with little available training data this transfer learning approach seems promising.

Next steps include

The TensorFlow checkpoint for our best model (3 frozen layers) is available online[7].

References

Agarwal, Aashish and Torsten Zesch (2019). "German End-to-end Speech Recognition based on DeepSpeech". In: *Proceedings of the 15th Conference on Nat-*

[7] ...

ural Language Processing (KONVENS 2019): Long Papers. Erlangen, Germany: German Society for Computational Linguistics & Language Technology, pp. 111–119.

Graves, Alex 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.

Hannun, Awni et al. (2014). *Deep Speech: Scaling up end-to-end speech recognition*. arXiv: `1412.5567 [cs.CL]`.

Heafield, Kenneth (July 2011). "KenLM: Faster and Smaller Language Model Queries". In: *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Edinburgh, Scotland: Association for Computational Linguistics, pp. 187–197. URL: `https://www.aclweb.org/anthology/W11-2123`.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.

Imai, Satoshi (1983). "Cepstral analysis synthesis on the mel frequency scale". In: *ICASSP'83. IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 8. IEEE, pp. 93–96.

Kingma, Diederik P. and Jimmy Ba (2014). *Adam: A Method for Stochastic Optimization*. arXiv: `1412.6980 [cs.LG]`.

Radeck-Arneth, Stephan et al. (2015). "Open Source German Distant Speech Recognition: Corpus and Acoustic Model". In: *Proceedings Text, Speech and Dialogue (TSD)*. Pilsen, Czech Republic, pp. 480–488.

Schröder, Marc and Jürgen Trouvain (2003). "The German text-to-speech synthesis system MARY: A tool for research, development and teaching". In: *International Journal of Speech Technology* 6.4, pp. 365–377.