Effects of Layer Freezing when Transferring DeepSpeech to New Languages

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 on German and Swiss German speech datasets and compare the results of different training methods. We first train the models from scratch on both languages and then improve upon the results by using an English pretrained version of DeepSpeech for weight initialization and experiment with the effects of freezing different layers during training. We see that even freezing only one layer already improves the results dramatically.

1 Introduction

The field of automatic speech recognition (ASR) is dominated by research specific to the English language. There exist plenty available text-to-speech models pretrained on (and optimized for) English data. When it comes to a low-resource language like Swiss German, or even standard German, the range of available pretrained models becomes very sparse. In this paper, we train Mozilla's implementation[1] of Baidu's DeepSpeech ASR architecture (Hannun et al. 2014) on these two languages. We use transfer learning to leverage the availability of a pretrained English version of DeepSpeech and observe the difference made by freezing different numbers of layers during training.

For previous work on using DeepSpeech for the two languages German and Swiss German, see (Agarwal and Zesch 2019) and (Agarwal and Zesch 2020) respectively. Note however, that our datasets and training methods are not identical to those used there. Our focus here lies on isolating the effect of layer freezing in the given context.

2 Transfer Learning and Layer Freezing

Deep neural networks can excel at many different tasks, but they often require very large amounts of training data and computational resources. To remedy this, it is often advantageous to employ transfer learning: Instead of initializing the parameters of the network randomly, the optimized parameters of a network trained on a similar task are reused. Those parameters can then be fine-tuned to the specific task on hand, using less data and fewer computational resources. In the fine-tuning process many parameters of the original model may be "frozen", i.e. held constant during training. This can speed up training, as well as decrease the computational resources used during training (Kunze et al. 2017). The idea of taking deep neural networks trained on large datasets and fine-tuning them on tasks with less available training data has been popular in computer vision for years (Huh, Agrawal, and Efros 2016). More recently, with the emergence of

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

end-to-end deep neural networks for automatic speech recognition (like DeepSpeech), it has also been used in this area (Kunze et al. 2017; Li, Wang, and Beigi 2019).

The reason why the freezing of parameters for fine-tuning deep neural networks is so successful, is that the networks learn representations of the input data in a hierarchical manner. The input is transformed into simplistic features in the first layers of a neural network and into more complex features in the layers closer to the output. With networks for image classification this can be nicely visualized (Zeiler and Fergus 2014). As for automatic speech recognition, the representations learned by the layers of a similar system to the one we used, one that is also based on Baidu's DeepSpeech architecture, have been analyzed by Belinkov and Glass (2017). The findings show that the hierarchical structure of features learned by DeepSpeech is not as clear as it is with networks for image processing. Nonetheless, some findings, for example that affricates are better represented at later layers in the network, seem to affirm the hypothesis that the later layers learn more abstract features and earlier layers learn more primitive features. This is important for fine-tuning, because it only makes sense to freeze parameters if they don't need to be adjusted for the new task. If it is known that the first layers of a network learn to identify "lower-level"-features, i.e. simple shapes in the context of image processing or simple sounds in the context of ASR, these layers can be frozen completely during fine-tuning.

The DeepSpeech network takes features extracted from raw audio data as input and outputs character probabilities (the architecture is described in more detail in the next section). With the reasoning from above, the first few layers should mostly obtain simple features, such as phonemes, from the input, while the later layers should mostly infer the character corresponding to these lower level features. The rationale for using transfer learning to transfer from one language to another, is the assumption that these lower-level features are shared across different languages. Thus, only the parameters later layers need to be adjusted for successfully training the network on a new language. Whether this assumption works in practice, and how much use freezing the layers actually is, will be the focus of this paper. We train the English pretrained version of DeepSpeech on German and on Swiss German data and observe the impact of freezing fewer or more layers during training.

3 Experimental Setup

3.1 DeepSpeech architecture

We use Mozilla's DeepSpeech version 0.7 for our experiments. The implementation differs in many ways from the original model presented by 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 probabilities for each character in the language's alphabet. It is fully connected and uses a softmax activation for normalization. The character-probabilities 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.

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

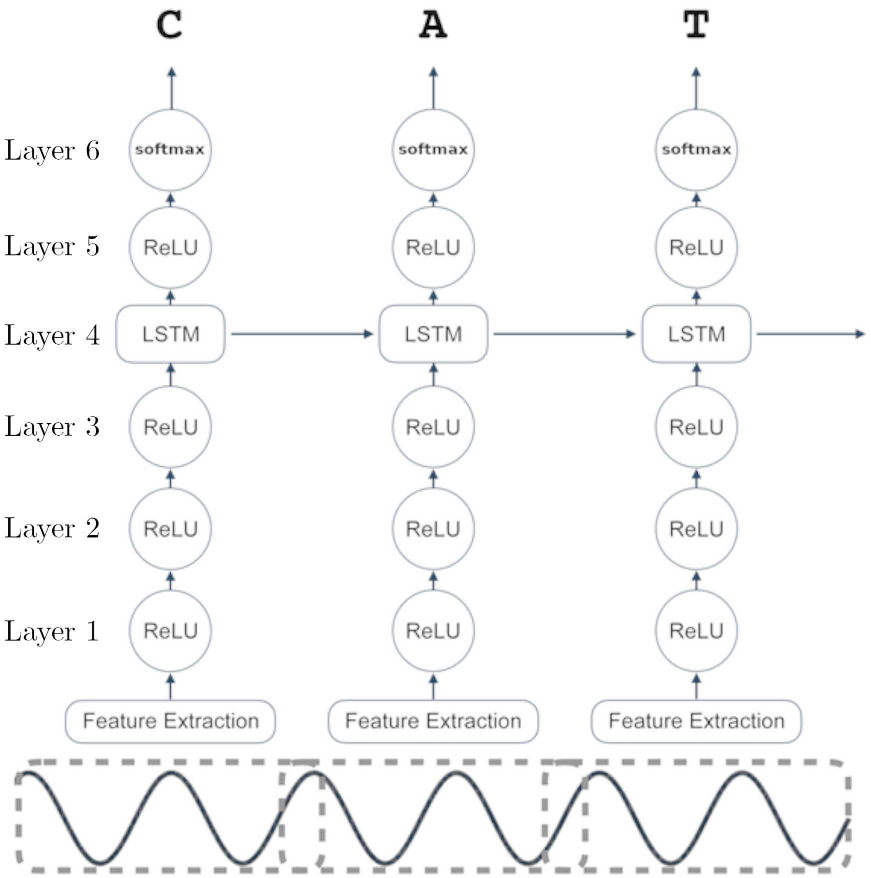

Figure 1: DeepSpeech architecture (adapted from the official documentation[2])

3.2 Training Details

To assess the effects of layer freezing, we train the network multiple times for each of the two languages. For weight initialization we use an English pretrained model, which is provided by Mozilla[3]. We then freeze between 0 and 4 layers during training. For both languages we also train one model from scratch, where the weights are initialized randomly. In total, we train 6 different models for each language:

Reference The whole model from scratch (random weight initialization)

0 Frozen Layers The model with weights initialized to those of the English pretrained model, all weights are optimized during training

1 Frozen Layer The English-initialized model with the first layer frozen

2 Frozen Layers The English-initialized model with the first two layers frozen

3 Frozen Layers The English-initialized model with the first three layers frozen

4 Frozen Layers The English-initialized model with the first three and the fifth layer frozen

The complete training script, as well as the modified versions of DeepSpeech that utilize layer freezing are available online[4]. The weights were frozen by adding `trainable=False` at the appropriate places in the TensorFlow code. For all models, we had to reinitialize

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

[4] `https://github.com/onnoeberhard/deepspeech-paper`

the last layer, because of the different alphabet sizes of German / Swiss German and English (ä, ö, ü).

3.3 Hyperparameters & Server

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 hyperparameter optimization. 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 the German language models for 30 epochs took approximately one hour per model. Training the Swiss German models took about 4 hours for 30 epochs on each model. We did not observe a correlation between training time and the number of frozen layers.

3.4 Datasets

We trained the German models on the German-language Mozilla Common Voice speech dataset (Ardila et al. 2020). The utterances are typically between 3 and 5 seconds long and are collected from and reviewed by volunteers. Because of this, the dataset comprises a large amount of different speakers which makes it rather noisy. The Swiss German models were trained on the data provided by Plüss, Neukom, and Vogel (2020). This speech data was collected from speeches at the Bernese parliament. The English pretrained model was trained by Mozilla on a combination of English speech datasets, including LibriSpeech and Common Voice English.[5] The datasets for all three languages are described in Table 1. For inference and testing we used the language model KenLM (Heafield 2011), trained on the corpus described by Radeck-Arneth et al. (2015, Section 3.2). This corpus consists of a mixture of text from the sources Wikipedia and Europarl as well as crawled sentences. The whole corpus was preprocessed with MaryTTS (Schröder and Trouvain 2003).

Dataset	Hours of data	Number of speakers
English	> 6500	—
German	315	4823
Swiss German	70	191

Table 1: Datasets used for training the different models

4 Results

The test results for both languages from the six different models described in Section 3.2 are compiled in Tables 2 and 3. For testing, the epoch with the best validation loss during training was taken for each model. Figures 2 to 5 show the learning curves for all training procedures (Fig. 2 and 3 for German, Fig. 4 and 5 for Swiss German). The curve of the best model (3 frozen layers for German, 2 frozen layers for Swiss German) is shown in both plots for each language. The epochs used for testing are also marked in the figures.

For both languages, the best results were achieved by the models with the first two to three layers frozen during training. It is notable however, that the other models that utilize layer freezing are not far off. The training curves look remarkably similar (see Figures 3 and 5). For both languages, all four models achieve much better results than the two models without layer freezing (“Reference” and “0 Frozen Layers”). The results seem to indicate that freezing the first layer brings the largest advantage in training, with

[5] See here for more detail: `https://github.com/mozilla/DeepSpeech/releases/tag/v0.7.0`

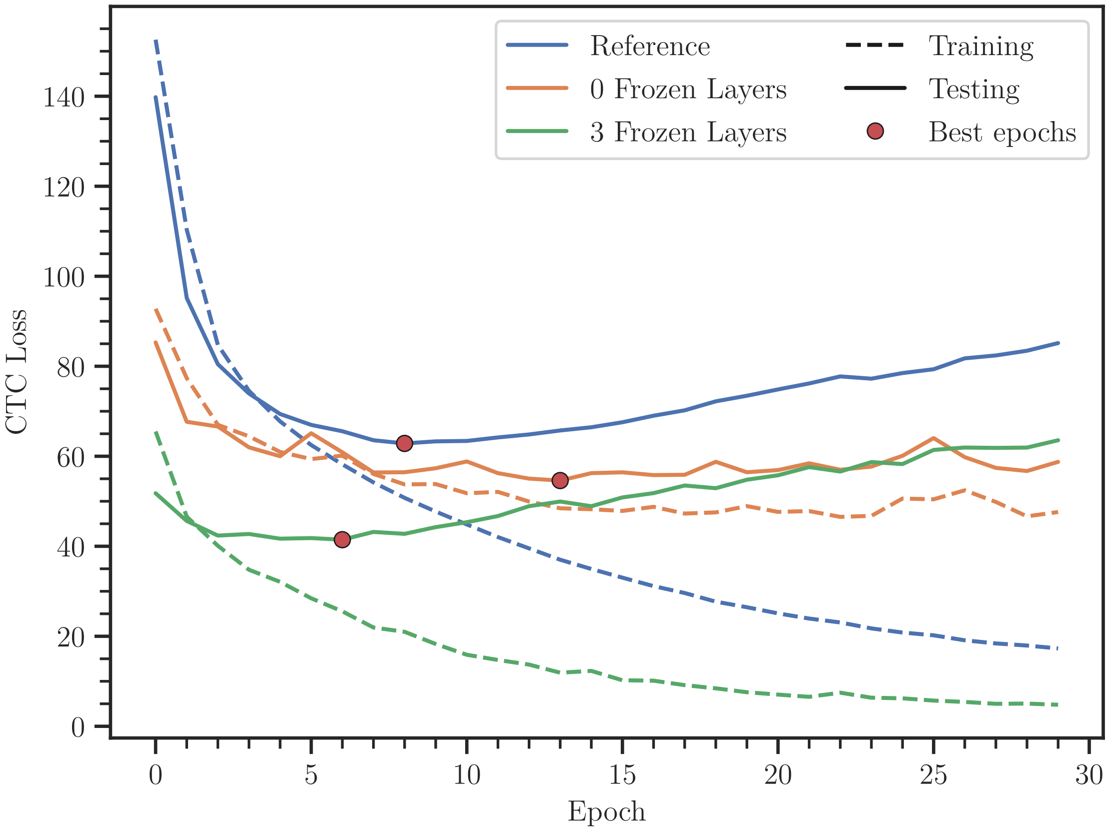

Figure 2: Learning curves (German dataset): With and without transfer learning and layer freezing

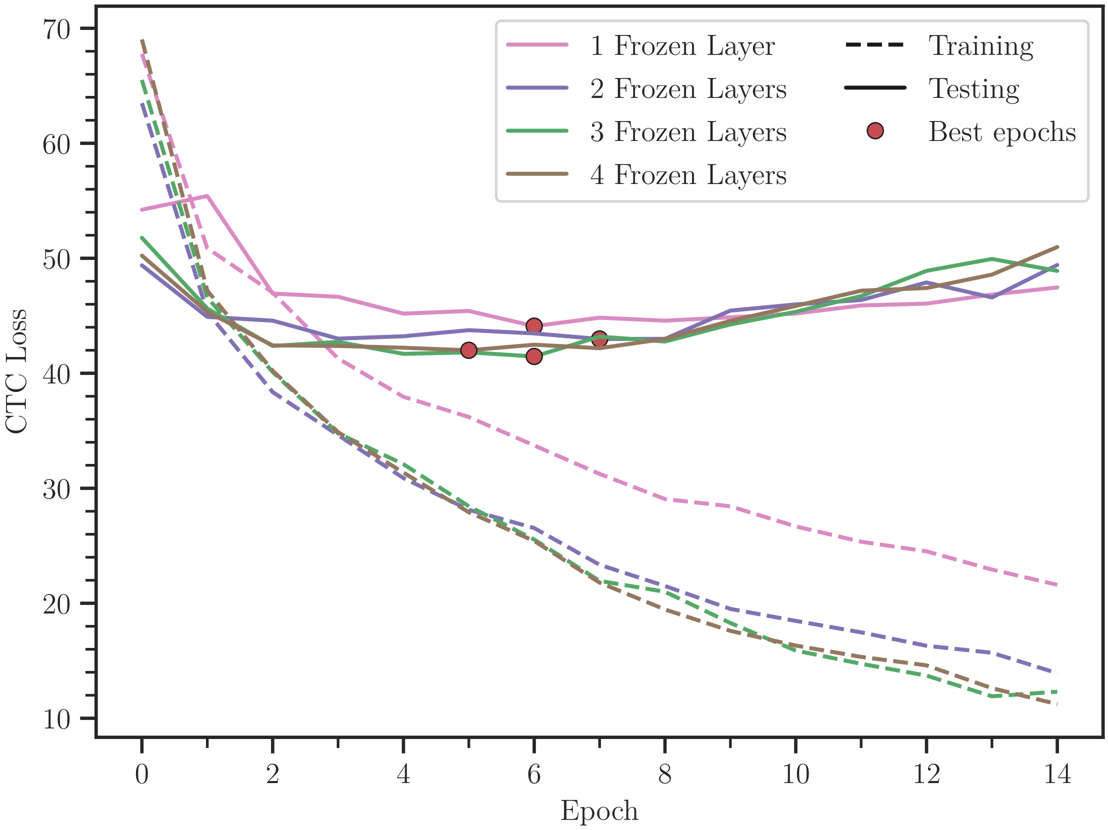

Figure 3: Learning curves (German dataset): Comparison of freezing a different number of layers

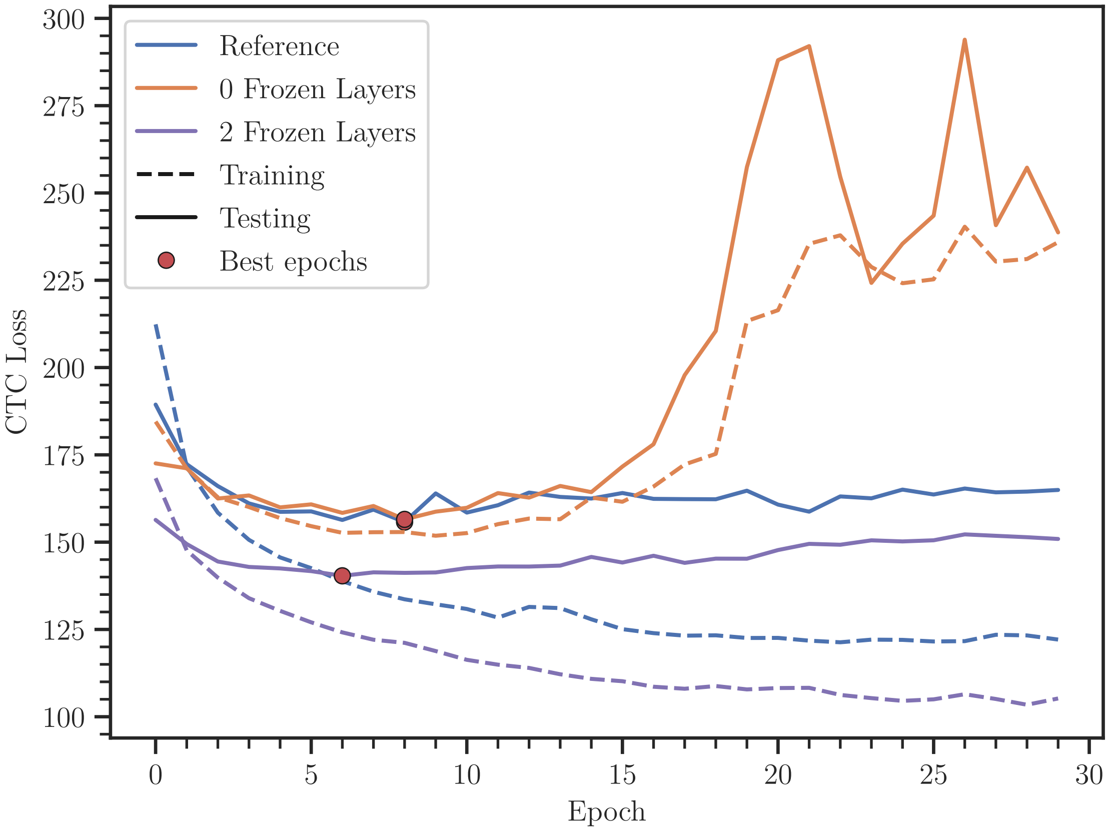

Figure 4: Learning curves (Swiss German dataset): With and without transfer learning and layer freezing

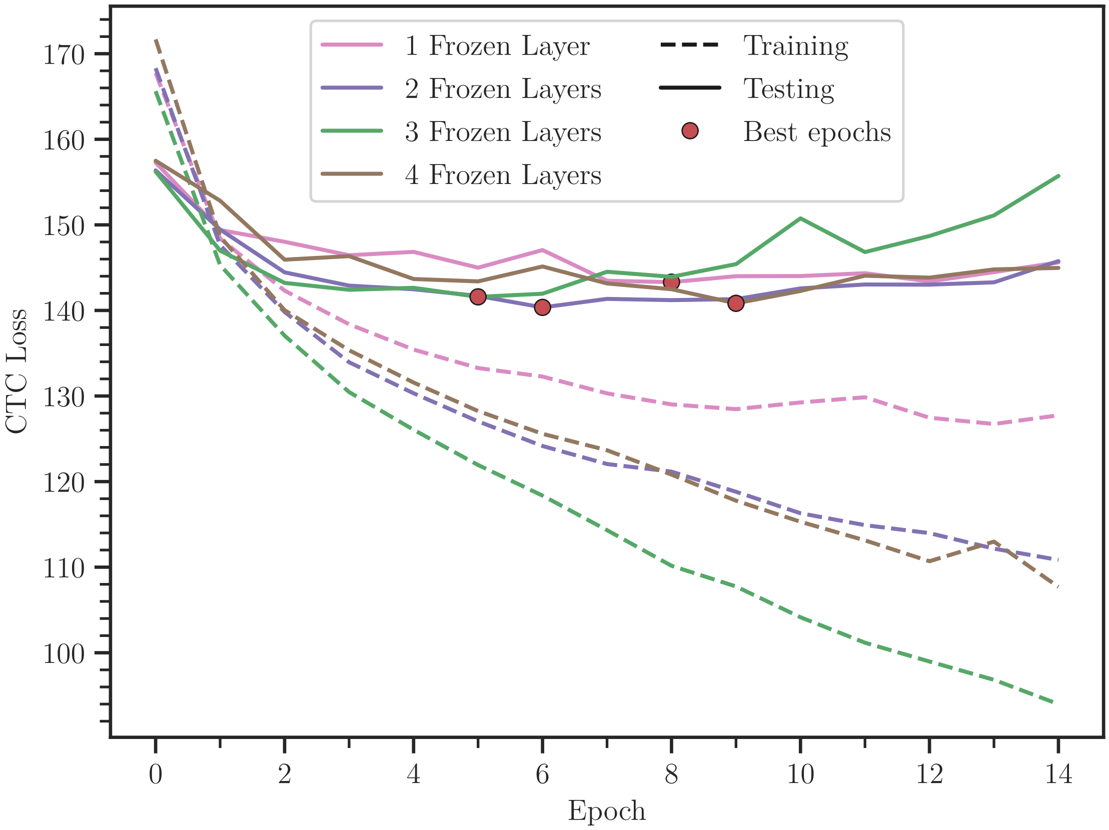

Figure 5: Learning curves (Swiss German dataset): Comparison of freezing a different number of layers

Method	WER	CER
Reference	.70	.42
0 Frozen Layers	.63	.37
1 Frozen Frozen Layer	.48	.26
2 Frozen Layers	**.44**	**.22**
3 Frozen Layers	**.44**	**.22**
4 Frozen Layers	.46	.25

Table 2: Testing results German

Method	WER	CER
Reference	.74	.52
0 Frozen Layers	.76	.54
1 Frozen Layer	.69	.48
2 Frozen Layers	**.67**	**.45**
3 Frozen Layers	.68	.47
4 Frozen Layers	.68	.46

Table 3: Testing results Swiss German

diminishing returns on freezing the second and third layers. For German, additionally freezing the fifth layer slightly worsens the result. For Swiss German, the result slightly worsens when the third layer is frozen and stays almost constant when additionally freezing the fifth layer. Similar results were achieved by Ardila et al. (2020), where freezing two or three layers also achieved the best transfer results for German, with a word error rate of 44%. They also used DeepSpeech and a different version of the German Common Voice dataset.

The models with four frozen layers could only optimize the LSTM weights and the weights of the output layer. It is surprising that they still achieve good results. It might be interesting to see what happens when the LSTM layer is frozen as well. 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. This might be the reason why with the larger German dataset the performance gets worse when freezing four instead of three layers, but not so with the smaller Swiss German dataset. For both languages it is evident that the transfer learning approach is promising.

5 Further Research

A next step might be to train these models with more training data and see if layer freezing is still beneficial. The chosen German speech dataset is not very large; Agarwal and Zesch (2019) achieved a best result of 0.151 WER when training the model on a large dataset, in contrast to a result of 0.797 WER when training the same model on a very similar dataset to the one we used.

An interesting idea for further research is to use a different pretrained model than the English one. English seems to work alright for transferring to German, but it is possible that the lower level language features extracted by a model only trained for recognizing English speech are not sufficient for transferring to certain other languages. For example, when just transcribing speech there is no need for such a model to learn intonation features. This might be a problem when trying to transfer such a pretrained model to a tonal language like Mandarin or Thai. There might also be phonemes that don't exist or are very rare in English but abundant in other languages.

6 Summary

Transfer learning seems to be a powerful approach to train an automatic speech recognition system on a small dataset. The effects we saw when transferring DeepSpeech from English to German and from English to Swiss German were very similar: The results were not necessarily better than plain training when just initializing the parameters, but freezing only the first layer already improved the results dramatically. Freezing more layers improved the outcome even more, but with larger training datasets this might have adverse effects.

7 Acknowledgements

We want to thank Aashish Agarwal for valuable help in setting up DeepSpeech and for providing preprocessing scripts as well as the hyperparameters we used for training.

References

Agarwal, Aashish and Torsten Zesch (2019). "German End-to-end Speech Recognition based on DeepSpeech". In: *Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers*. Erlangen, Germany: German Society for Computational Linguistics & Language Technology, pp. 111–119.

— (2020). *LTL-UDE at Low-Resource Speech-to-Text Shared Task: Investigating Mozilla DeepSpeech in a low-resource setting.*

Ardila, Rosana et al. (2020). "Common Voice: A Massively-Multilingual Speech Corpus". In: *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*. European Language Resources Association, pp. 4218–4222.

Belinkov, Yonatan and James Glass (2017). "Analyzing Hidden Representations in End-to-End Automatic Speech Recognition Systems". In: *Advances in Neural Information Processing Systems*. Vol. 30, pp. 2441–2451.

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 (2011). "KenLM: Faster and Smaller Language Model Queries". In: *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, pp. 187–197.

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

Huh, Minyoung, Pulkit Agrawal, and Alexei A. Efros (2016). *What makes ImageNet good for transfer learning?* arXiv: `1608.08614 [cs.CV]`.

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]`.

Kunze, Julius et al. (2017). "Transfer Learning for Speech Recognition on a Budget". In: *Proceedings of the 2nd Workshop on Representation Learning for NLP, Rep4NLP@ACL 2017, Vancouver, Canada, August 3, 2017*. Association for Computational Linguistics, pp. 168–177.

Li, Bryan, Xinyue Wang, and Homayoon S. M. Beigi (2019). "Cantonese Automatic Speech Recognition Using Transfer Learning from Mandarin". In: *CoRR*.

Plüss, Michel, Lukas Neukom, and Manfred Vogel (2020). *Germeval 2020 task 4: Low-resource speech-to-text.*

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.

Zeiler, Matthew D. and Rob Fergus (2014). “Visualizing and Understanding Convolutional Networks”. In: *Computer Vision – ECCV 2014*. Springer International Publishing, pp. 818–833.