Self-supervised pretraining for low resource languages in Speech processing
Project Report

Nicolas Dufour
ENS Paris Saclay
nicolas.dufourn@gmail.com

Julien Hauret
ENS Paris Saclay
julien.hauret@ens-paris-saclay.fr

Abstract

In October 2020, a paper entitled "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations" was published by A.Baevski et al. from Facebook AI [1]. It claims to achieve 1.8/3.3 WER on the clean/other test sets of Librispeech in English. What is even more impressive is the performance when using just ten minutes of annotated speech: 4.8/8.2 WER. The goal here is to use the pretrained features of wav2vec 2.0 with a small network and a CTC[1] loss to perform a phone recognition system in new languages. We will pay special attention to the amount of data used and the proximity between languages. The latest multilingual pretrained version of wav2vec 2.0 which is XLSR-53 [2] will also be used and compared.

1. Introduction and Motivation

Modern machine learning approaches and especially deep learning have been a game changer is many fields including ASR [2]. This is due to the fact that the downstream task in performed in an end-to-end to fashion which allows to capture all the relevant information. Nevertheless, the complex architecture of these methods often includes convolutions and transformers which need a lot of audio annotated data and computational power to be trained at their full potential. As a consequence, it is not always possible to reach state-of-the-art results because of the restrictive amount of available data and limited computational power. From those observations, we want to use a previously trained models on specific languages as a phone recognition on other untrained languages. Our aim is then to assess its capability to re-use its learned features.
We only implemented this phone recognition framework instead of an entire ASR system because the language model required to pass from one to another is acting independently.

[1]Connectionist Temporal Classification
[2]Automatic Speech Recognition

2. Related work

The Deep Speech 2 paper [3], published in 2016 was a game changer in ASR world. It proved that neural networks overpass all previous methods when sufficient amount of audio annotated data is available. However, this is not always the case. To deal with this lack of data, self-supervised learning approach and in particular CPC[3] was proven to be very effective. It circumvents the issue by predicting the future in latent space by using powerful autoregressive models. This concept was introduced the paper [4] in 2018. The huge network which is wacv2vec 2.0 [1] used this trick to pretrain. The other notable difference is that it starts from raw audio rather than previously used spectrogram. Finally, the XLSR-53 [2] makes the most of the wav2vec 2.0 architecture and the multilingual concepts introduced in [5].

3. Model and Methodology

Our experiments consist in re-using/fine-tuning the XLSR-53 and wav2vec 2.0 models to produce phoneme recognition system in different languages and verify or refute some hypothesis.

3.1. Model

The XLSR-53 architecture is the same as the wav2vec 2.0. It only differs from the training set which is multilingual in the first case and monolingual in the second. This architecture is a powerful (3.5 Go to store the weights of the network) combination of convolutions and transformers. It takes in input the raw waveform and first passes trough multi-layer convolutions to produce a latent speech representation. The transformer network follows suit to build a contextualized representation of the speech. Finally, we add a classic fully connected layer to map this representation to the phoneme vocabulary by giving a degree of confidence for every element.

[3]Contrastive Predictive Coding

3.2. Loss

The loss is the Connectionist Temporal Classification as described in [6]. This loss is particularly adapted for speech recognition where the labels are not aligned with the audio files. This loss allows to asses the ability of the network to produce a good phonemization by maximizing the probability of all the valid paths.

3.3. Evaluation Metric

We evaluate our models according to the PER[4] . The PER is computed according to the phonemes Levenshtein distance which computes the number of edits, insertions and deletions to go from one phoneme sequence to an other. Lower is better.

3.4. Our training procedure

3.4.1 Warm-up

We observed that our training was very sensible and had trouble to solve the optimisation problem. To help with the training, we decided to use a learning rate warm-up with linear decay. The idea is that for the first 2000 steps, we increase the learning rate from zero to the maximum learning rate. Then we slowly decay the learning rate until reaching again zero at the end of the training. The idea here is to first not get stuck in a local minimum at the beginning of the training, and then, we want our system to converge more precisely to the best model. We use a maximum learning rate of 3e-4.

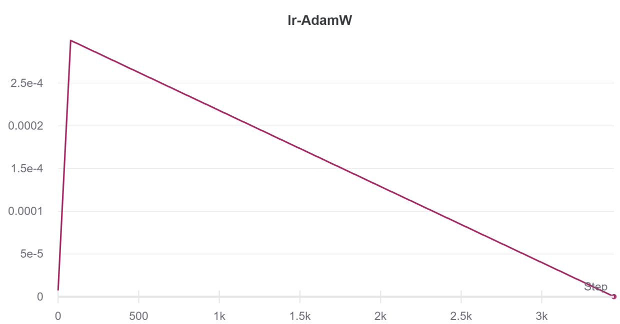

Figure 1. Learning rate scheduler

3.4.2 Optimizer

We use an AdamW optimizer with a weight decay parameter of 0.01. AdamW is known to be better at handling weight decay than Adam resulting in more reliable training.

3.4.3 The plateau phenomenon

As we can see on Figure 2, the loss first dive and then stay on a plateau for a few epochs. When the model is on the plateau it only produces blanks. If we keep training, it overpass the limit and reach a better optimum.

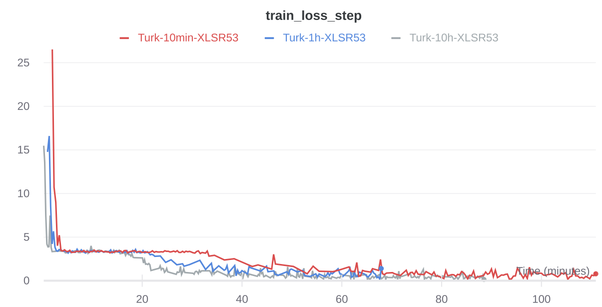

Figure 2. CTC Training curves of our model

After the plateau the PER begins to decrease as we can see on Figure 3.

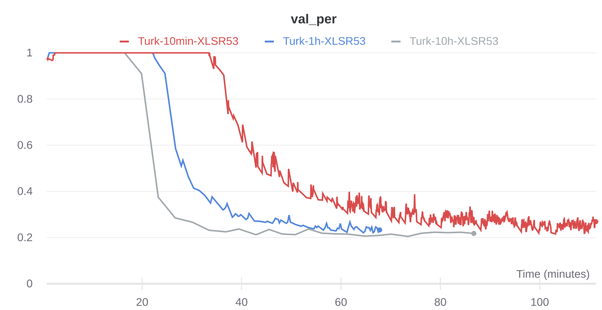

Figure 3. PER Validation curves of our model

This made the finetuning of this model hard. Indeed, we had numerous experiment fail because we didn't manage to cross that plateau.

3.5. Data

We will indirectly use the Librispeech (LS-960) [7] dataset and the large LibriVox (LV-60k) unlabelled dataset by using the pretained models available on [8].
The multilingual model XLSR-53 combines 3 datasets for pretraining: MLS [9], CommonVoice [10] and Babel [11].
To train our CTC network we have chosen some labelled datasets from the Mozilla's common voice project website [12].

4. Results

4.1. Compare training with 10min/1h/10h of data

The results of our experiments is represented in the Table 1. We have chosen languages that appear in the pre training of the XLSR-53 network in different proportions. For languages that XLSR-53 have seen only few examples or even none, the behaviour is logic:

$$PER_{10min} > PER_{1h} > PER_{10h}$$

[4] Phonemes Error Rate

However, languages that have been used in a consequent proportion for the pretraining of the XLSR-53 model (with the contrastive task) have unexpected results. Indeed, the ordering of the PER seems to be random or even reverse. An attempt to explain this strange behaviour is that the pretrained model is already good on some languages that it already seen in huge proportions and our fine tuning is just harming its performances.

	Proportion in XLSR-53	10min	1h	10h
Greek	None	0.3099	0.2312	0.1353
Turk	Low	0.4226	0.3172	0.2263
French	High	0.1514	0.2700	0.2875
Spanish	High	0.1401	0.2866	0.2232

Table 1. Test PER for different amount of audio annotated data for different languages

4.2. Comparison between wav2vec 2.0 (English) and multilingual XLRS-53

In this part we want to compare two Wav2Vec2 models. One is pretrained on English and the other on multiple languages (XLSR-53). The idea here is to try to understand what does training language diversity brings to the table when trying to train on a new language. We explicitly choose to test on two languages that weren't part of XLSR-53 training set, Greek and Czech. Both language are relatively low ressource with the Greek having 13h of data and the Czech having 45h of data. We also train on a Wav2vec2 model that has not been pretrained to have a baseline and try to understand what does pretraining brings. The results can be seen in Table 2

	English	Multilingual	No pretraining
Greek	0.1227	0.1301	0.8775
Czech	0.1022	0.1136	0.9311

Table 2. Test PER for different pretraining models

We can see that pretraining has a major impact on the performances of the model. The model fails to learn anything meaningful without pretraining. To our surprise, the English pretraining actually works better than the multilingual pretraining.

4.3. Study of the link between languages proximity and fine-tuning

On this section, we want to study the importance of language proximity for fine-tuning. Looking at Figure 4, we see that some languages are closer than others. We decide to test this fine-tuning on two languages (Portuguese and Dutch), on two pretrained models (Spanish and German). Portuguese and Dutch have 63 hours of data. Indeed,

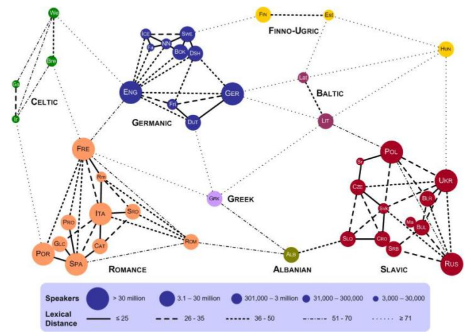

Figure 4. Proximity between languages by [13]

Portuguese and Spanish are closer languages in the same manner than Dutch and German are closer languages. We need to disclose that our experimental setup isn't perfect. Indeed we use pretrained models that actually have been fine tuned on XLSR-53. The model has actually already seen the languages we want to test. Also, pre-training on already fine-tuned models isn't actually optimal since the model features might have been skewed towards the language we fine-tuned on, making it harder to fine-tune again. To be more precise, we would need to train models on only Spanish and only German. We however didn't have the resources to do that.

	Pretrained German	Pretrained Spanish
Portuguese	0.1506	0.1307
Dutch	0.2518	0.2502

Table 3. Test PER for different languages depending on language proximity with pretrained language

As we can see in Table 3, It is indeed easier to fine-tune Portuguese on a Spanish model than on a German model. The results aren't that obvious for the Dutch fine-tuning.

5. Conclusion

In this work we have studied the impact of pretrained models for speech recognition systems. Some languages have a lot of data available which allows to train massive models that generalize well. However, for most languages, there is not this data available at all. We must rely on self supervised learning and models pretrained on other languages. This work study the transfer capabilities of Wav2Vec2 models and in particular XLSR-53, which trains on multiple datasets and multiple languages.

We have shown that decent performances are reachable when training on very small datasets with 10min/1h/10h of data. We observed a correlation that for languages that weren't very present in the XLSR-53 training set, we have higher performance with more data. Surprisingly, we observe the inverse behaviour when using dataset that were more present on the training set.

We also proved the utility of using pretrained models over random initialized ones. However, we did not observe any improvement when using a model trained on English data or a multilingual one.

Finally, we observed that there is an improvement when training on a model that has been pretrained on data from a language that is similar to the language we want to fine tune on.

6. Further Work

We would want to have a more in-depth study of the comparison of Wav2Vec2 English and XLSR-53. Our results goes against our intuition and previous results from [2]. We think that maybe the learning rate we choose might not be appropriated for all languages and we might need to search for better hyper-parameters to fine tune our models. The models we use are very sensible to the training procedure and it is hard to find the best hyper-parameters.

In general training was quite hard. We wonder if a better training procedure could lead to more stable and reliable results.

7. Task sharing

7.1. Work done by Nicolas Dufour

Nicolas fully implemented the `CommonVoiceDataModule` class inside which he performed the phonemization of annotated sentences and a clean implementation of Pytorch Lightning [14] DataLoader that can directly be used for training. This implementation works fine for every language dataset from CommonVoice. He also implemented the PER[5] function using PytorchMetrics. Finally, he had a global view of the project when training some models.

7.2. Work done by Julien Hauret

Julien makes use of the Pytorch's CTCloss function in the class `CTCNetwork` that he implemented using also Pytorch Lightning, making it compatible with the Nicolas' DataLoader. One attribute of the class is the wav2vec 2.0 model that he used with the framework of Hugging Face Library [15]. Just as Nicolas, he also trained some models on different languages.

8. Code

The code can be found here: https://github.com/nicolas-dufour/self-unsupervised-low-res-speech.

[5] Phone error rate

References

[1] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *arXiv preprint arXiv:2006.11477*, 2020. 1

[2] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, "Unsupervised cross-lingual representation learning for speech recognition," *arXiv preprint arXiv:2006.13979*, 2020. 1, 4

[3] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, pp. 173–182, PMLR, 2016. 1

[4] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018. 1

[5] M. Rivière, A. Joulin, P.-E. Mazaré, and E. Dupoux, "Unsupervised pretraining transfers well across languages," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7414–7418, IEEE, 2020. 1

[6] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376, 2006. 2

[7] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 5206–5210, IEEE, 2015. 2

[8] Pytorch, "wav2vec." https://github.com/pytorch/fairseq/blob/master/examples/wav2vec. 2

[9] V. Pratap, Q. Xu, A. Sriram, G. Synnaeve, and R. Collobert, "Mls: A large-scale multilingual dataset for speech research," *arXiv preprint arXiv:2012.03411*, 2020. 2

[10] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," *arXiv preprint arXiv:1912.06670*, 2019. 2

[11] "Babel program." 2

[12] "Mozilla's common voice project." https://commonvoice.mozilla.org/fr. 2

[13] T. Elms, "Lexical distance among the languages of europe. etymologikon march 4: 2008," 2008. 3

[14] W. Falcon and .al, "Pytorch lightning," *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, vol. 3, 2019. 4

[15] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, (Online), pp. 38–45, Association for Computational Linguistics, Oct. 2020. 4