Active Learning and Task-Adaptive Pre-Training for Low-Resource Language Transcription

Meredith Xu *
Department of Computer Science
Stanford University

Claire Zhang *
Department of Electrical Engineering
Stanford University

Abstract

Developing a deep learning-based automatic speech recognition system (ASR) that performs well has been an active research area in the speech community, particularly for low resource languages where it is difficult to obtain large amounts of transcribed text. Previous work has shown that using task-adaptive pre-training (TAPT) along with active learning can reduce the training data needed for several NLP tasks such as sentiment analysis and topic classification. In this project, we examine whether and to what extent TAPT and active learning with entropy sampling can help reduce the training data required to achieve comparable result as using a model without task-adaptive pre-training fine-tuned with 100% of the training data for ASR task using a Dutch dataset. We discover that TAPT lowers the word error rate (WER) of the predicted transcriptions by half compared to the model without pre-training on target dataset for all dataset sizes, but active learning with entropy sampling does not outperform random sampling.

1 Introduction

Automatic speech recognition is a task that takes audio clips as inputs and outputs a hypothesized text transcription corresponding to each audio clip. Manually transcribing speech is a time-consuming and labor-intensive task (CaseGuard). For languages that are not commonly spoken by a lot of people, it is hard to obtain sufficient amounts of transcribed data that can be used to train a deep learning automatic speech recognition system (ASR). Hence, Building an ASR system that performs well still remains a challenging task, especially for low resource languages. The goal of this project is to examine how models pre-trained on unlabeled data can be used to minimize the amounts of transcribed data (labeled data) required during fine-tuning when developing an ASR system. In particular, we use pre-trained wav2vec2 models (Baevski et al., 2020) (Bartelds and Wieling, 2022) and examine the effects of domain adaptation and active learning.

The goal of domain adaptation is to allow a model to perform well on test dataset that is of a different domain than the training dataset. Task-adaptive pre-training (TAPT) is a specific domain adaptation method where a model, which is pre-trained with self-supervision on unlabeled data of a non-target dataset, is further trained with self-supervision on unlabeled data of the target dataset and later fine-tuned with supervision on labeled data of the target dataset. Gururangan et al. (2020) demonstrates that domain adaptation through TAPT can lead to performance gains. Active learning is a training approach where during each training iteration, the model is fed a subset of data that it is least certain of, which can be measured by entropy. Previous works show that this approach can reduce the amount of training data required to achieve comparable result as using all the training data (Margatina et al., 2022; Varadarajan et al., 2009).

The goal of this project is to evaluate whether domain adaptation and active learning can help pre-trained wav2vec2 models achieve performance gains in speech recognition, with indication of potential uses of these methods for low-resource language transcription. Experiment results demonstrate that TAPT greatly improves the model's performance while active learning with entropy sampling does not consistently help boost the model results. We propose a few hypothesis of why active learning does not help in our case.

*Equal contribution, alphabetical by last names.

Proceedings of the Spring 2022 offering of Computer Science Two Hundred and Twenty Four S, Spoken Language Processing, Stanford, California, USA.

2 Related Work

2.1 wav2vec2 for ASR

Built on self-supervised learning, wav2vec2 (Baevski et al., 2020) is a framework that learns robust representations from raw audio data, in the form of a finite set of speech units shorter than the phonemes. As shown in Figure 1, wav2vec2 first processes raw audio waveforms through a multi-layer convolutional neural network (CNN) to learn latent audio representations. The representations are then fed to a transformer to learn a contextualized representation, and to a quantizer to choose a specific speech unit for this representation from a pool of already learned units. About half of the representations are masked out before being fed to the transformer. The model is trained to distinguish true latent units from distractors via a contrastive task. After pre-trained on the unlabeled data, the model is then fine-tuned using labeled data with a Connectionist Temporal Classification loss (Graves et al., 2006). Baevski et al. (2020) states that using only 10 minutes of labeled data for fine-tuning, this approach achieves word error rate 4.8%/8.2% on the clean/other test sets of Librispeech and 1.8%/3.3% word error rate when fine-tuning using all 960 hours of labeled data from Librispeech.

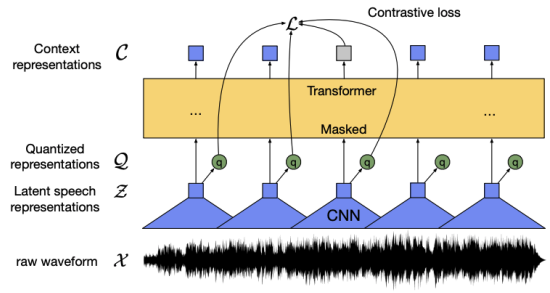

Figure 1: Architecture of wav2vec 2.0

2.2 TAPT

It is foundational to pre-train language models on a large corpus of unlabeled text data from various sources. Built on this premise, Gururangan et al. (2020) finds that additional pre-training with domain adaptation to in-domain data improves the performance, and further pre-training with task adaption improves the performance even more. As mentioned in the introduction, TAPT is a method where prior to any supervised fine-tuning on the target dataset, a pre-trained model is further trained in a self-supervised manner on the unlabeled target dataset. Gururangan et al. (2020) shows that domain-adaptive pre-training and task-adaptive pre-training offer performance gains for several domains in text classification tasks, both in low resource and high resource settings.

We investigate the effect of TAPT in helping large ASR models transcribe low-resource languages. We examine whether applying TAPT with a language that has limited transcribed text (e.g. Dutch) on a model pre-trained on a different language (e.g. English) will achieve performance gains over the model without TAPT.

2.3 Active Learning

Active learning refers to an iterative training method where during training the system can query a human annotator to label a subset of data, e.g. utterances whose transcriptions the system is most uncertain of (Cohn et al., 1994; Settles, 2009). The subset of data annotators are able to label out of all the unlabeled data is the fixed labeling cost, which is also referred to as the fixed annotation budget (Weng, 2022). Acquisition functions are used to sample the new set of data that will be added to the training data for the next active learning iteration. Commonly used sampling strategies are uncertainty sampling and diversity sampling (Weng, 2022). One example of the acquisition functions is entropy that measures the uncertainty of the model's predictions (Dagan and Engelson, 1995). Data with higher entropy will be sampled to the next iteration's training pool for the model to train on. Margatina et al. (2022) demonstrates that using active learning in combination with TAPT when adapting pre-trained language models for NLP tasks can help reduce the reliance on labeled data to as little as 15% to achieve the same performance as training with 100% of the data (baseline). This work also shows that the model adaptation strategy is more important than the acquisition functions.

In this project, we examine whether and to what extent this approach can help reduce reliance on transcribed speech for adapting a pre-trained speech model for the task of ASR on simulated low-resource conditions.

3 Dataset

We use the Corpus of Spoken Dutch (CGN) dataset,[1] which is a collection of 900 hours of Dutch speech. Dutch dataset was chosen because TAPT is extremely resource intensive and out of scope for this course project and the model pre-trained on Dutch already exists. To simulate low resource conditions, we have randomly sampled 5 hours of data (3 hours training, 1 hour dev, 1 hour test) from the read speech component (243 hours) of this dataset. The sampled data is roughly balanced in terms of the metadata of the speakers such as gender and has no overlap in speakers between train, test, and validation sets. The data has already been pre-processed for us and contains audio clips and their transcribed texts. Figure 2 shows the distribution of length of all audio files in our dataset. We can see that most of the audios are on the longer end. The longest audio is about 30 seconds and the shortest is about 0.938 second.

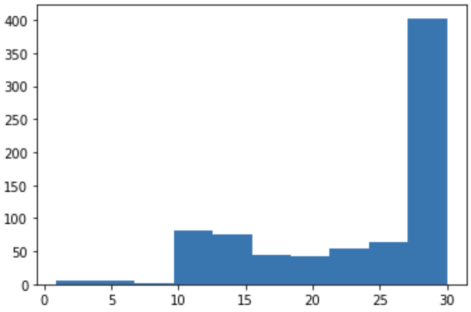

Figure 2: Audio File Length Distribution (x axis is audio duration in seconds and y axis is number of audio files)

We match the characters that appear in our dataset with the vocabulary of the Dutch pre-trained and fine-tuned model to make sure that our training vocabulary is the same as the model's vocabulary. There are a total of 37 characters in the model vocabulary [2]. Those are the commonly used characters in Dutch.

4 Approach

Among many wav2vec2 variants pre-trained on different datasets, we use the 'LS960' monolingual English variant trained on 960-hour LibriSpeech corpus (LS960) as our off-the-shelf pre-trained model (Baevski et al., 2020) (wav2vec2 large) [3].

We use the above off-the-shelf model that is further pre-trained on a 243-hour subset of the CGN Dutch dataset (LS960+CGN243, prepared by: Bartelds and Wieling, 2022) as our task-adapted pre-trained model (wav2vec2 large Dutch).[4] Both models are available on HuggingFace. We take those models and fine-tune them with the 3-hour CGN Dutch data described in the Dataset section (drawn from the 243-hour CGN subset used to adapt the original LS960 model).

We apply active learning with entropy sampling during fine-tuning and compare that with random sampling. Entropy measures the level of uncertainty over probabilities and entropy sampling allows us to select samples that the model is most uncertain of. To evaluate the uncertainty of the model's prediction on a sample, we compute the entropy over the probabilities of different characters at every time step for a transcription and obtain an average over the entire utterance (Varadarajan et al., 2009). We can get the probabilities of different characters at every time step directly from the model's output. The equation for calculating entropy of one prediction is as follows:

$$Entropy = -\sum_{y} P_\theta(y|x) * log(P_\theta(y|x))$$

. We evaluate the model from the previous iteration on the candidate data pool and select samples assigned with higher entropy scores into the training pool for the model's next round of fine-tuning. For random sampling, instead of selecting samples with higher entropy into the next training pool, we assign each sample with a random score drawn from the standard normal distribution and pick samples with higher scores to the next iteration. We also fine-tune the off-the-shelf wav2vec2 large model and further pre-trained Dutch wav2vec2 large model with all the training data. They serve as baselines to compare with the models that use active learning.

The code for fine-tuning and evaluating wav2vec2 models for ASR has already been made available,[5] and we wrote the code for different sampling strategies and ran sets of experiments. The link to the code can be found in the Appendix A. The models we trained are shown in Table 1.

[1] https://taalmaterialen.ivdnt.org/download/tstc-corpus-gesproken-nederlands/

[2] https://huggingface.co/GroNLP/wav2vec2-dutch-large-ft-cgn/blob/main/vocab.json

[3] https://huggingface.co/facebook/wav2vec2-large

[4] https://huggingface.co/GroNLP/wav2vec2-dutch-large

[5] https://github.com/CoEDL/vad-sli-asr

Model
No TAPT + 100% Train Data (Baseline)
TAPT + 100% Train Data (Baseline)
No TAPT + Random Sampling
No TAPT + Active Learning
TAPT + Random Sampling
TAPT + Active Learning

Table 1: Trained Model Summary

5 Experiments

5.1 Metrics

The metrics we use is the word error rate (WER) and character error rate (CER). We train and evaluate each one of the models described above and compare their WER/CER. WER and CER show similar trends, so the discussions on the results made later in the paper are based on WER. By comparing the WER/CER produced by different models, we can analyze roughly how long it takes (number of training steps) for each model to converge on WER/CER and evaluate if active learning and TAPT would require less data to converge.

5.2 Active Learning Setting

For each of the active learning experiments (non-baselines), we run 5 iterations of active learning and 10% of new data is added to the training data pool for each active learning iteration. At the beginning of each iteration, we draw 10% of the total training data from an unlabeled pool of data using either entropy or random sampling. Therefore, for 5 active learning iterations, we have 10%, 20%, 30%, 40%, and 50% of total data respectively and the models are trained from the beginning for each iteration using the data it has acquired so far. Note that the first 10% data for the first iteration is always randomly sampled. We stop at 50% due to the limited time we have for our experiments, which is generally referred to as fixed annotation budget and is a common practice used in active learning.

5.3 Fine-tuning Setting

We fine-tune all the models with 2000 optimization steps and run evaluation on the development set every 100 steps. We also implement early stopping which would stop the training if after 600 steps, the model's WER does not drop for 4 consecutive evaluations. We use AdamW optimizer (Loshchilov and Hutter, 2017) with learning rate warm-up steps of 200. We also have learning rate of 1e-5 and batch size of 16 with gradient accumulation steps of 2. We chose these hyperparameters by experimenting with different hyperparameters on models fine-tuned with only 10% data. All the experiments were run with the same hyperparameters and each run took around 3 hours on average. With these hyperparameters, all the models we trained were able to converge on a decent WER. One thing we found during initial hyperparameter tuning is that for ASR models with little data (e.g. 10% data), it is hard to get good performance without doing extensive hyperparameter search, even with TAPT models. The training curves for all the experiments can be found in the Appendix A.

5.4 Result

Figures 3, 4, 5, and 6 show results of fine-tuning wav2vec2 (non-TAPT) and wav2vec2 Dutch (TAPT) models for 5 active learning rounds with random sampling or entropy sampling.

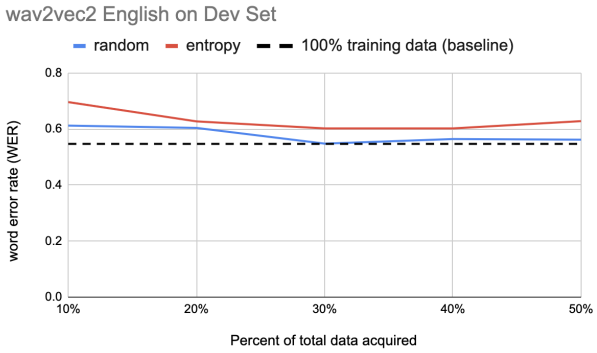

Figure 3: WER comparison of random sampling, entropy sampling, and the baseline for non-TAPT model on the development set

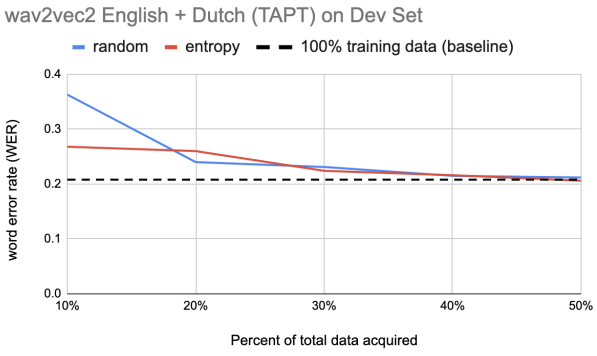

Figure 4: WER comparison of random sampling, entropy sampling, and the baseline for TAPT model on the development set

Comparing figure 5 and 6, we can see that TAPT model produces a WER that is almost half of the

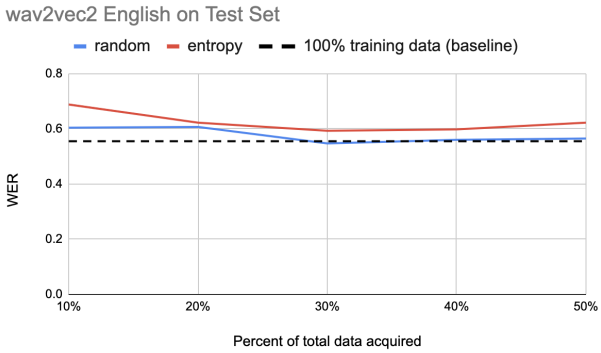

Figure 5: WER comparison of random sampling, entropy sampling, and the baseline for non-TAPT model on the test set

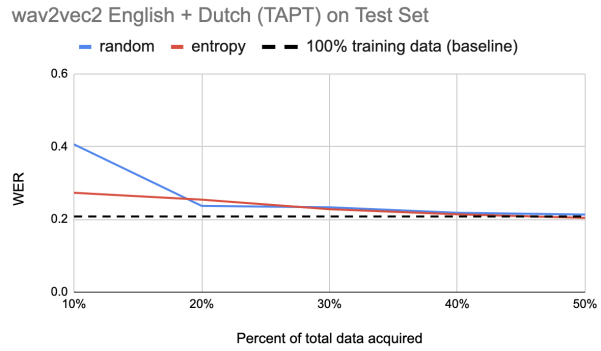

Figure 6: WER comparison of random sampling, entropy sampling, and the baseline for TAPT model on the test set

model without TAPT for all dataset sizes, indicating that using a model pre-trained on the task specific data the model is later fine-tuned on can greatly improve the model's performance for ASR task. However, active learning with entropy sampling does not show the same effect. In figure 5, we can see that for wav2vec2 model (non-TAPT), random sampling actually performs better than entropy sampling. Figure 6 shows that for the TAPT model, the comparison between random and entropy sampling is more variable and entropy sampling does not consistently outperform random sampling.

Notably, two of our fine-tuning runs (wav2vec2 model with entropy sampling on 10% data and wav2vec2 Dutch model with random sampling on 10% data) crashed without finishing the whole training process due to a Pytorch memory issue. The wav2vec2 model with entropy sampling on 10% data crashed at 1000 steps and the wav2vec2 Dutch model with random sampling on 10% data crashed at 900 steps. That is why for both models, after first iteration (i.e. training with 10% data), there is a gap between the WER produced by random sampling and entropy sampling. Theoretically, we would expect the models to output similar WER after the first iteration for both random sampling and entropy sampling since we randomly sample the data in the first iteration for both models. Initially, we hypothesized that the crash at first iteration for wav2vec2 with entropy sampling could result in an underfitted model and thus affect the entropy calculation results, which in turn could have an impact on the new samples that get selected for the second iteration. To investigate whether the crash actually affected the entropy experiment result at the second iteration, we took the fine-tuned wav2vec2 model with random sampling after first iteration and performed entropy sampling with it to obtain the new set of training samples. Then, we trained this model for one iteration of active learning. The resulting model actually had a slightly higher WER than the fine-tuned wav2vec2 model with entropy after second iteration from our original experiment (figure 7), which indicates that the crash was not the reason that entropy sampling did not perform better than random sampling. Overall, our experiment shows that random sampling and entropy sampling have similar performances for our task. We hypothesize that one reason active learning with entropy sampling does not outperform random sampling could be that ASR task is more challenging than natural language processing tasks that do not involve speech data. Hence, it might be beneficial to first feed the model some easier examples to learn by starting with several iterations of random sampling and then continuing with entropy sampling, instead of directly starting with hardest examples which is what active learning with entropy sampling does.

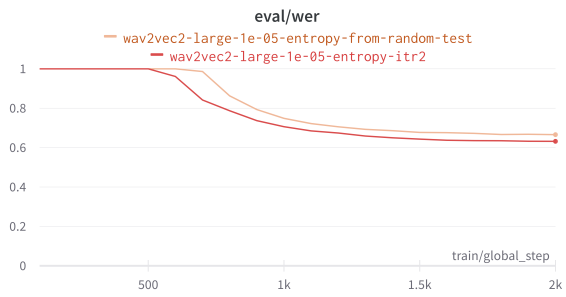

Figure 7: red line is the fine-tuned wav2vec2 model with entropy after second iteration from the original experiment and orange line is the model that starts with wav2vec2 model with random and then performs entropy sampling

To further investigate why entropy sampling did not help with model performance, it would be interesting to look at the specific examples assigned with high entropy values that are selected to the next training pool. The two sentences assigned with the highest entropy values in each sampling round by the TAPT model are shown in the following table. The first sampling round is not in the table because it uses random sampling.

wav2vec2 English + Dutch (TAPT)

Iteration	Transcription	Entropy
20%	HOOFDSTUK ˌjäˌjäN	0.1112
	DE AFVAART WOENSDAG TIEN APRIL NEGENTIENHONDERD TWAALF	0.1098
30%	HOE LAAT MOETEN WE MORGENOCHTEND WEG? VROEG IK OOM HARRY HAALDE Z'N ZAKHORLOGE TE VOORSCHIJN EN KNIPTE 'T OPEN ALS WE ZE MORGEN BEGRAVEN ZEI HIJ ZWAARWICHTIG ZAL IK JE OM KWART OVER ZES MOETEN AFHALEN ZODAT WE HEEN EN TERUG NAAR JERSEY CITY KUNNEN RIJDEN	0.1426
	PRINCIPIˌjäLE DOORGETHEORETISEERDE FILOSOFISCHE HOPELOOSHEID DE WHITMANIAANSE HOOP DIE DE HARTEN VAN AMERIKAANS LINKS VOOR DE JAREN ZESTIG DEED OPSPRINGEN WORDT NU BESCHOUWD ALS EEN SYMPTOOM VAN NAˌjèEF HUMANISME IK ZIE DEZE VOORKEUR VOOR KENNIS BOVEN HOOP ALS EEN HERHALING VAN DE ZET DIE LINKSE INTELLECTUELEN DEDEN TOEN ZE EERDER DEZE EEUW HET HEGELIANISME VIA MARX EN NIET VIA DEWEY TOT ZICH NAMEN MARX DACHT DAT WE WETENSCHAPPELIJK MOESTEN ZIJN IN PLAATS VAN ALLEEN MAAR UTOPISCH	0.1337
40%	GEUREN TEER EN TOUW DAT MOETEN BIJNA ZEKER DE EERSTE GEUREN ZIJN GEWEEST DIE MIJN VADER ROOK VERS NIEUW TOUW ZEILDOEK EN TEER	0.1403
	HIJ VRAAGT VERDER MAAR NIETS 'T HEEFT TOCH GEEN ZIN OOIT HEEFT IE GEVRAAGD HOE DIE ZOU HETEN ALS IE EEN MEISJE WAS GEWEEST PAPA WIST NOG WELKE MEISJESNAMEN ZE HADDEN BEDACHT ANNA ELLE EN AKKA	0.1371
50%	DAT ZE DE TWEE HONDJES BIJ ZICH HIELDEN IN HUN HUTTEN ANDERE PASSAGIERS VAN DE TITANIC HADDEN ERVOOR GEKOZEN DE HONDEN IN DE KENNEL VAN HET SCHIP ONDER TE BRENGEN HAD HIJ GEHOORD TERWIJL ZIJN MOEDER TOEZICHT HIELD OP HET UITPAKKEN VAN DE BAGAGE BESLOOT BILLY OM MET DE KING CHARLESSPANIˌjäL HET DEK OP TE GAAN	0.1344
	ALS GOD DAAR DEEL VAN UITMAAKTE TOT JE DIENST EEN HOGERE MACHT EEN LIEFHEBBEND WEZEN EEN MACHT ACHTER NA NA NATUURWETTEN BEST	0.1341

Figure 8: Examples with top two entropy in each iteration for TAPT model.

Some special characters in the transcription text might be caused by the file saving software Microsoft Excel not being able to properly process Dutch specific characters. We do not speak Dutch, but by consulting our mentor Martijn Bartelds who speaks Dutch, we do observe that sentences with high entropy values tend to contain proper names and many long Dutch words that are infrequently used in common Dutch vocabulary, which makes these examples more difficult to transcribe. This could be a reason that entropy sampling doesn't give higher WER than random sampling. Those observations are made just by glancing over the examples. However, to dive deep into investigating why certain sentences are more difficult, it would be helpful for a Dutch speaker to look over those examples in greater details.

6 Conclusion

We study the effects of task-adaptive pre-training and active learning for speech recogonition task where there is a limited amount of labeled data using pre-trained wav2vec2 large models. We find that with only 10% of labeled data for fine-tuning a TAPT model, we can lower the WER by half over fine-tuning with 100% of data on a non-TAPT model. Such finding suggests that using pre-trained models can reduce the amounts of data needed to achieve great performance in ASR task, which can help reduce the human transcription burden in practice. We also observe that active learning alone does not offer a consistent performance boost over models without active learning.

Given the challenging nature of the ASR task, one way to extend our work is to experiment with other ways of data sampling methods, such as curriculum learning (Kuznetsova et al., 2022) or a mixture of random and entropy sampling. We could also try out different acquisition functions in active learning and see if they would give different results than entropy sampling. Due to the time and resource constraints, we only ran each experiment once. In the future, we will repeat each experiment multiple times (e.g. 3 times) to get more consistent results.

7 Acknowledgement

This work is done in collaboration with our mentors Nay San, Martijn Bartelds, and Katerina Margatina. We greatly appreciate their guidance and mentorship for this project.

8 Contribution

Meredith Wrote script for running experiment and active learning. Ran experiments and analyzed the crashed runs. Also did paper writing.

Claire Wrote script for running experiment and active learning. Ran experiments. Also did paper writing.

Our mentor Nay San prepared the fine-tuning script and Martijn Bartelds prepared our dataset.

References

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations.

M. Bartelds and M. Wieling. 2022. Quantifying language variation acoustically with few resources. *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

CaseGuard. Automatic transcription vs manual transcription. time, effort, and cost.

David Cohn, Les Atlas, and Richard Ladner. 1994. Improving generalization with active learning. *Mach. Learn.*, 15(2):201–221.

Ido Dagan and Sean P. Engelson. 1995. Committee-based sampling for training probabilistic classifiers. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 150–157. Morgan Kaufmann.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 369–376, New York, NY, USA. Association for Computing Machinery.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Anastasia Kuznetsova, Anurag Kumar, Jennifer Drexler Fox, and Francis Tyers. 2022. Curriculum optimization for low-resource speech recognition.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Katerina Margatina, Loic Barrault, and Nikolaos Aletras. 2022. On the importance of effectively adapting pretrained language models for active learning. *arXiv preprint arXiv:2104.08320*.

Burr Settles. 2009. Active learning literature survey.

Balakrishnan Varadarajan, Dong Yu, Li Deng, and Alex Acero. 2009. Maximizing global entropy reduction for active learning in speech recognition.

Lilian Weng. 2022. Learning with not enough data part 2: Active learning. *lilianweng.github.io*.

A Appendices

The code we wrote for this project can be found at `https://github.com/fauxneticien/active_learning-w2v2_asr`.

The following figures show the WER, CER, and loss during training for both wav2vec2 (non-TAPT) and wav2vec2 Dutch (TAPT) models with entropy sampling and random sampling for each iteration. The run names have the following format: (model name)-(learning rate)-(acquisition method)-(iteration number).

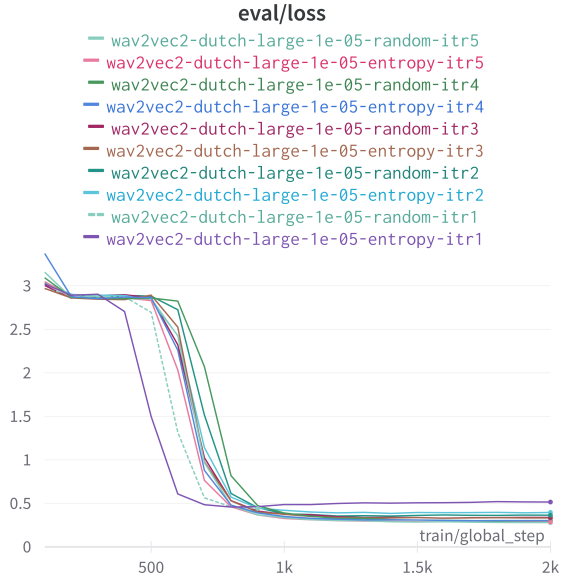

Figure 9: Training loss curves for TAPT model with random and entropy sampling at each iteration

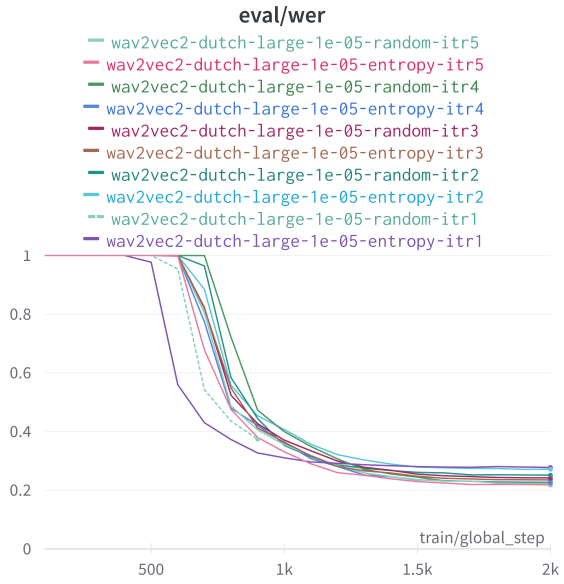

Figure 10: Training WER curves for TAPT model with random and entropy sampling at each iteration

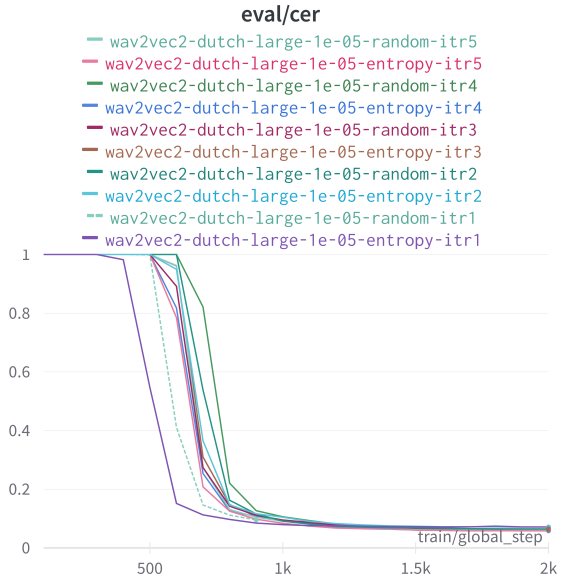

Figure 11: Training CER curves for TAPT model with random and entropy sampling at each iteration

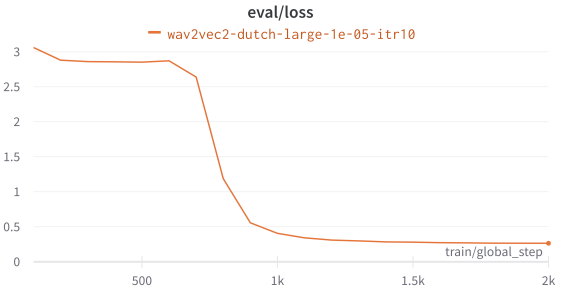

Figure 12: Training loss curve for baseline TAPT model trained on 100% data

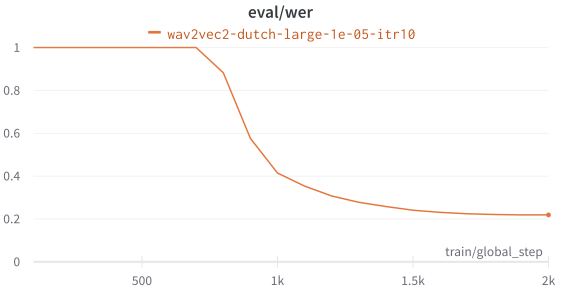

Figure 13: Training WER curve for baseline TAPT model trained on 100% data

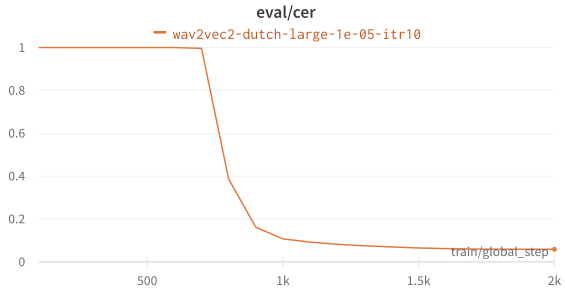

Figure 14: Training CER curve for baseline TAPT model trained on 100% data

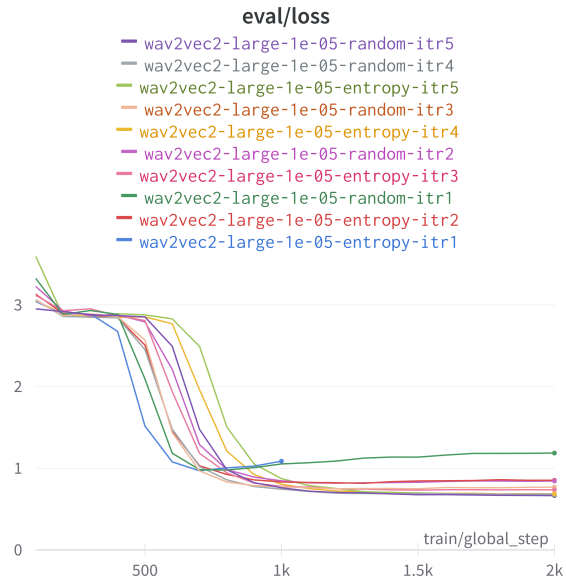

Figure 15: Training loss curves for non-TAPT model with random and entropy sampling at each iteration

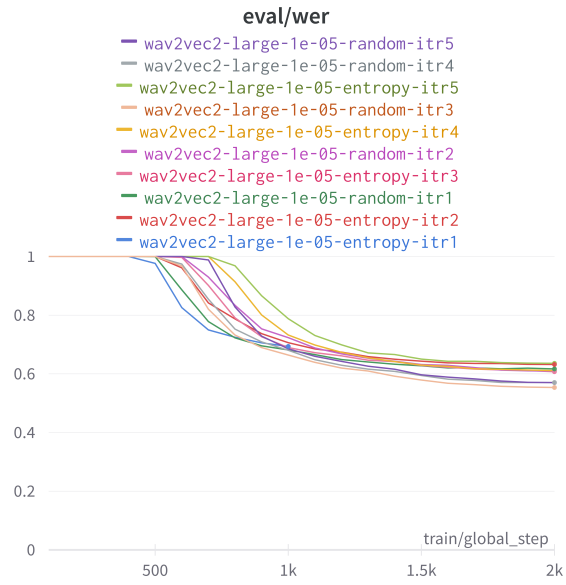

Figure 16: Training WER curves for non-TAPT model with random and entropy sampling at each iteration

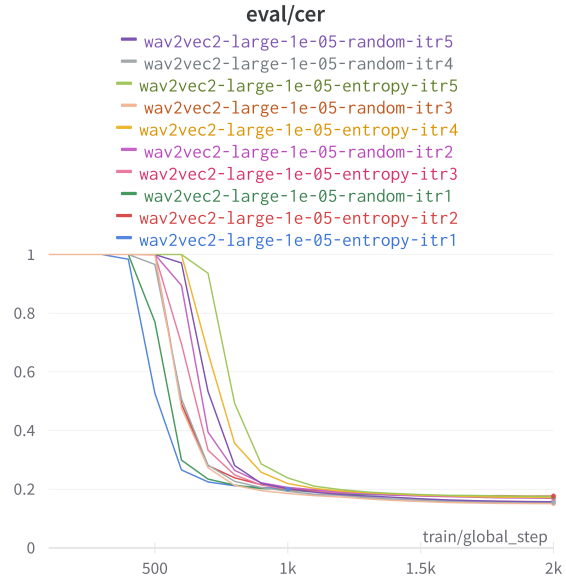

Figure 17: Training CER curves for non-TAPT model with random and entropy sampling at each iteration

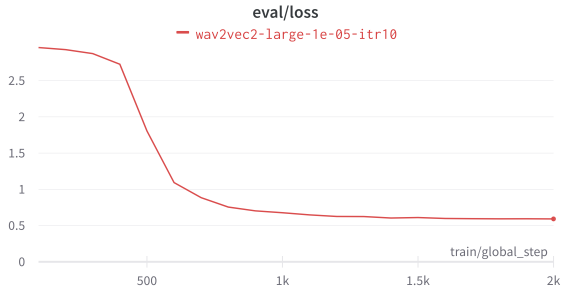

Figure 18: Training loss curve for baseline non-TAPT model trained on 100% data

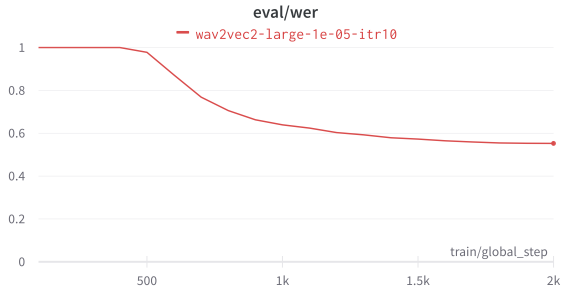

Figure 19: Training WER curve for baseline non-TAPT model trained on 100% data

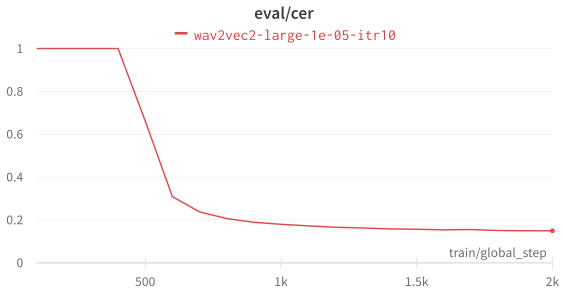

Figure 20: Training CER curve for baseline non-TAPT model trained on 100% data