Methods to Optimize *Wav2Vec* with Language Model for Automatic Speech Recognition in Resource Constrained Environment

Vaibhav Haswani
vaibhavhaswani@gmail.com

Padmapriya Mohankumar
padmapriya.mohankumar@gmail.com

Abstract

Automatic Speech Recognition (ASR) on resource constrained environment is a complex task since most of the State-Of-The-Art models are combination of multilayered convolutional neural network (CNN) and Transformer models which itself requires huge resources such as GPU or TPU for training as well as inference. The accuracy as a performance metric of an ASR system depends upon the efficiency of phonemes to word translation of an Acoustic Model and context correction of the Language model. However, inference as a performance metric is also an important aspect, which mostly depends upon the resources. Also, most of the ASR models uses transformer models at its core and one caveat of transformers is that it usually has a finite amount of sequence length it can handle. Either because it uses position encodings or simply because the cost of attention in transformers is actually $O(n^2)$ in sequence length, meaning that using very large sequence length explodes in complexity/memory. So you cannot run the system with finite hardware even a very high-end GPU, because if we inference even a one hour long audio with Wav2Vec the system will crash. In this paper, we used some state-of-the-art methods to optimize the Wav2Vec model for better accuracy of predictions in resource constrained systems. In addition, we have performed tests with other SOTA models such as Citrinet and Quartznet for the comparative analysis.

1 Introduction

Speech is the most natural way of human communication; it gives humans a medium to understand and communicate their feelings and emotions. Understanding speech or speech recognition is a critical part of modern applications even plays a significant role for empowering AI enabled smart devices such as Amazon's Echo, Google's Nest, Apple's Homepod etc.

In this paper, we have proposed a method called `WSLR` (Wav2Vec with Stride Chunking and Language Model for Resource-constrained devices) to create an offline speech recognition system using Wav2Vec model which achieved a Word Error Rate (WER) of 0.85 in an environment of 2 core CPU and 4 gigabytes of RAM on Librispeech test set, which surpasses the score of original Wav2Vec model i.e. 1.8 and other SOTA models. To recognize speech of an audio file we first start off by converting it into a digital format since audio data could have a lot of variations like different channels or sample rate. We further have must standardize the dataset by re-sampling the data to a specific sample rate as any model requires data to follow common standards. We have used Wav2Vec-base model trained on `librispeech` data for 960 hours as our main ASR model for the speech recognition task where it encodes sound files via multi-layer convolutional neural network (CNN) to produce latent speech representation and then feed those masked representations to a transformer network, the output of which can be decoded to word vectors using a Connectionist Temporal Classification (CTC) algorithm. The following model can further be optimized when the output logits are fed to a language model (in this paper we used a 4-gram language model). (Baevski et al., 2020) [1] To deploy our model to a resource constrained environment we use something called *stride chunking*, here the chunking is sequential, and we chunk the audios such that there is some overlapping length at borders defined by 'stride length' and the main context stay intact within the center. Then while inference, overlapped segments or logits (as output) are dropped to keep the results of the transcriptions as similar as they would be while inferenced with full audio. We also use some rule-based post-processing steps for our transcription to generalize even better for spoken numbers or symbols in-case.

[1] stride chunking reference at `https://huggingface.co/blog/asr-chunking`

Proceedings of the 19th International Conference on Natural Language Processing (ICON), pages 149 - 153

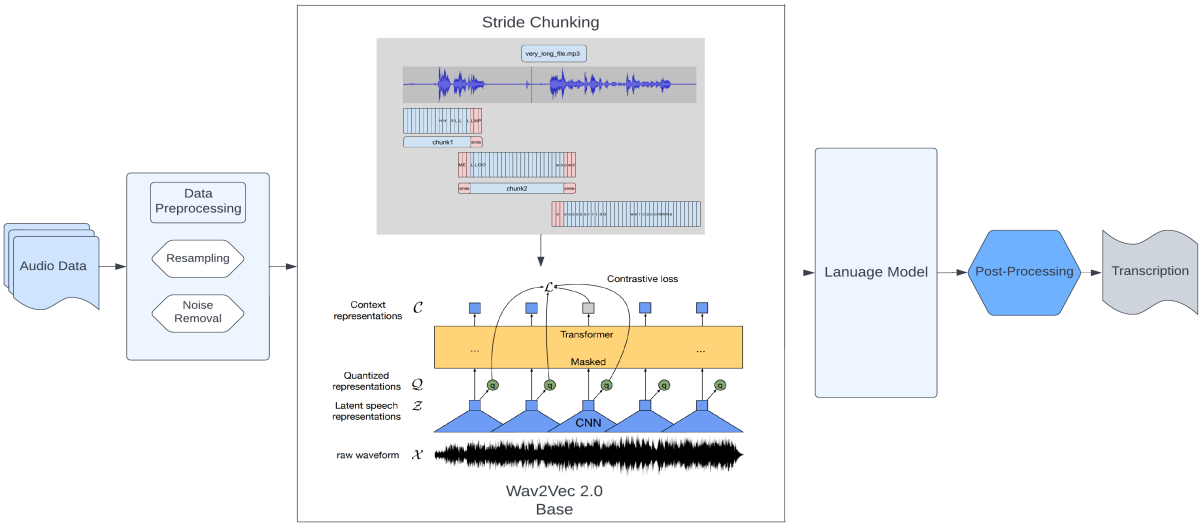

Figure 1: Proposed workflow of optimized Wav2Vec

2 Proposed Method

In this section, we discuss the proposed approach i.e. WSLR to optimize the state-of-the-art wav2vec model introduced by Facebook and that achieves such an optimal performance when inferenced in constrained environments such as, constrained docker container or embedded/micro systems like Raspberry Pi or Nvidia's Jetson board. Constrained inference along with good scores is quite a big deal for a huge deep learning model. Since, more the size of the model the more resources it demands and Wav2Vec is itself a combination of huge models such as CNNs and Transformers. Therefore, it is quite a challenging task. The slight drop in scores while chunking is comprehended with pre-processing, language model, and post-processing optimizations. Further, the details of the optimization methods and proposed functionalities are discussed in the following sub-sections.

2.1 Data Processing

Data Processing of audios aims to standardize the audios with a uniform setting, for evaluations we have used `librispeech` test-clean set by (Panayotov et al., 2015), and for explicit data pre-processing we have used `ffmpeg` utility for conversion of media to standard .wav format and `librosa` package for loading and re-sampling files. We first off defined pre-processing module to convert files to standard wav format and then re-sample it to 16KHz since Wav2Vec 2.0 strictly targets audios sampled at the mentioned rate. Text pre-processing methods such as, lower-casing, removal of special case characters and unwanted white spaces are performed on the librispeech test transcripts.

2.2 Stride Chunking

One way to inference from a transformer-based model for long audios in constrained environment could be by chunking the audio in parts of equal length (Mauranen and Vetchinnikova, 2017) and send those chunks to the model iteratively and later generate the whole prediction by aggregating all the sub predictions. This is computationally efficient but usually leads to subpar results. A major caveat of this kind of chunking is poor context at the border of chunks, i.e., since model requires previous context to generate better transcriptions this kind of chunking would give a non-contextual part of speech to the model which would lead to poor transcription.

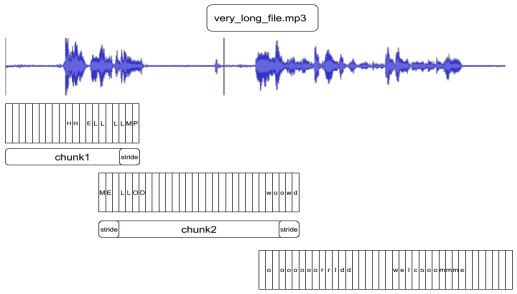

Figure 2: Reference for stride chunking from huggingface

Stride Chunking is even a smarter way of chunking audios inspired by stride concept in CNNs where we chunk the audios by leaving some overlapping frames from previous and next chunk, this way model will have proper context in the center for the inference. In stride chunking, while inference overlapped segments or logits are dropped after inference to generate aggregated output of all chunks i.e. proper transcription as we get it from the whole audio file. In our experiments we have tested different hyper-parameters and found that 30 seconds of `chunk length` with 2 seconds of `stride length` from right and 1 second of stride from left are giving optimal results in terms of inference and recognition quality.

2.3 ASR Model - Wav2Vec Base 960h

As an ASR model we used state-of-the-art model by Facebook-AI i.e., Wav2Vec 2.0 base 960h (we also used Wav2Vec 2.0 large in our experiments but selected the base model since it performs better for constrained environments). Proposed in (Baevski et al., 2020), "Wav2Vec is composed of a multi-layer convolutional feature encoder f : X $\rightarrow$ Z which takes as input raw audio X and outputs latent speech representations z1, . . . , zT for T time-steps. They are then fed to a Transformer g: Z $\rightarrow$ C to build representations c1, . . . , cT capturing information from the entire sequence [9, 5, 4]. The output of the feature encoder is discretized to qt with a quantization module Z $\rightarrow$ Q to represent the targets in the self-supervised objective (§ 3.2). It also builds context representations over continuous speech representations and self-attention captures dependencies over the entire sequence of latent representations end-to-end." the model is further improved with Hidden-Unit BERT (HuBERT) approach proposed by (Hsu et al., 2021) and XLS-R for cross lingual speech representation by (Babu et al., 2021).

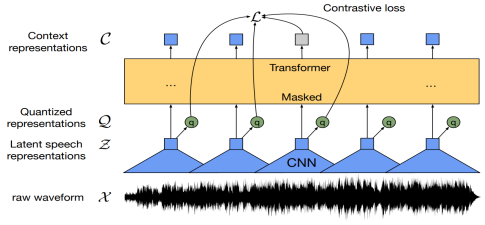

Figure 3: Wav2Vec architecture by Facebook AI

2.4 Language Model

A Language Model (LM) captures how words are typically used in a language to construct sentences or paragraphs. It could be a general-purpose model about a language such as English or Japanese, LM could be used to predict the next word in a sentence, to discern the sentiment of some text. Previously audio classification models required an additional LM and a dictionary to transform the sequence of classified audio frames to a coherent transcription based on context. Wav2Vec's architecture is based on transformer layers, thus having the context in consideration it can produce coherent transcriptions even without a language model. In addition, Wav2Vec2 leverages the CTC algorithm for fine-tuning, which solves the problem of alignment between a varying "input audio length"-to-"output text length" ratio. In conclusion, Wav2Vec model doesn't require a separate language model to generate acceptable transcripts. However, the performance of the model can be boosted by integrating a separate LM model to optimize the predicted transcript results even further. The language model should be good at modeling language that corresponds to the target transcriptions of the speech recognition system. In our experiments we have trained three language models via `kenlm` library Heafield (2011) i.e., 3-gram , 4-gram and 5-gram Language model and have used a 4-gram LM since it yields better scores in terms of WER. The model is trained to generalize the occurrence of maximum four consecutive words at a time also the model happens to correct a lot of word sequence errors of the Wav2Vec model transcription, as given in Table 1.

Table 1: Language Model Benchmarks

n-gram model	WER
3-gram	0.85122
4-gram	0.85095
5-gram	0.85097

2.5 Post-processing and Final Transcriptions

Post-Processing steps may consist of various methods like deep learning models or rule-based approaches such as Inverse Text Normalization (ITN) that is the task of converting the raw spoken output of the ASR model into its written form to improve text readability. Post-Processing is an important step to generalize transcriptions for readability and

understanding. for e.g., spoken text is "I was born in nineteen eighty" or "my email is robert at the rate transformers dot com" in both the example transcriptions are not as generalized and would not look appealing in terms of readability, and in applications where a real-time speech to text system is been utilized such as Zoom meetings those kind of transcriptions won't be appreciated. So, once we apply ITN to the transcriptions the quality of the transcript readability increases significantly, let's take the previous examples and apply ITN to them, the output will be like - "I was born in 1980" and "my email is robert@transformers.com", here we can observe that the transcriptions are more generalized in terms of numbers and symbols. Similarly other post-processing approaches are such as Auto Punctuation , where we mostly use a deep Neural Network (DNN) to understand the context of the raw transcription and generate richer transcriptions with punctuation or capitalization based on context of the raw text. ASR systems typically generate texts without punctuation or capitalization, and punctuation can add an ability to understand the meaning of the text, where whole meaning of the sentence can change with a slight change in punctuation, that is the power this small post-processing step holds. Applying these post-processing methods produce even richer and readable transcriptions which makes the pipeline more robust and scalable.

3 Experiments and Comparative Analysis

This section presents the experimental evaluations of the optimized approach, using Wav2Vec model with optimization methods mentioned in the former sections. For the experiments, we have used Docker Containers to reflect the constraints of mobile devices and the experiments are performed with 2 most common mobile configurations i.e., 2 cores CPU with 8 gigabytes of RAM and 2 cores CPU with 4 gigabytes of RAM. It has additionally been tested with a GPU configuration comprised of Tesla P100 for a better intuition of the performance. We have done a cross-comparative analysis of the system proposed with general wav2vec model inference as well as other state-of-the-art models like Quartznet by (Kriman et al., 2020) and Citrinet model by (Majumdar et al., 2021).

For the experiments, we used librispeech's test set and pre-processed it further to generalize it for metric calculations. Table 3, presents the benchmarking runs performed on different resource configurations and it can be observed that even though on a RAM restricted environment such as 4GB RAM environment with 2 core CPU our system did not crash and gave almost similar inference to the environment with 8GB of RAM.

Table 2: A summary of the experimental settings

Environment	**Configurations**
Machine/CPU	Intel Haswell
GPU	Tesla P100
Operating system	Ubuntu 20.04 LTS
Docker Image	Python Slim 3.9
Memory Config (RAM)	8 GB/4GB
CPU Config (Cores)	2
Neural network library	`PyTorch`

Table 3: Benchmarks on different resource configurations

Resource Config	**WER**	**Inference (s)**
2 CPU, 8GB RAM	0.85	6381
2 CPU, 4GB RAM	0.85	6456
1 GPU, 8GB RAM	0.85	356

Along with WER we have calculated Jaccard Similarity scores of predicted transcripts which is another metric for text similarity and the pipeline tends to achieve a jaccard score of 0.92 on the same librispeech test set. For the comparative analysis we have compared the results of our approach with different state of the art methods. Table 4, presents the comparative tests based on librispeech clean test set between different state-of-the-art models, it can be observed that the proposed method Word Error Rate is significantly lower when compared to other common ASR models and with lower WER we get more accurate predictions. A detailed comparison of other non auto-regressive models are presented in (Ng et al., 2021).

Table 4: Librispeech test set benchmarks on different models

Model	**WER**
Proposed Work	**0.85**
Nvidia Quartznet	3.78
Wav2Vec Large	1.8
Nvidia Citrinet	2.7

4 Conclusion

This paper presents, an optimized way of scaling offline ASR system to resource constrained environments, in our experiments we have noted a very significant improvement such as 0.85 of WER and pipeline inference on environment with 4 gigabytes of RAM and 2 CPU cores without facing any system crashes also the results on resource constrained environment are mostly similar to a high-end resource environment, some related work of speech recognition in low end systems is also presented by (Thomas et al., 2013) and (Bansal et al., 2018). The proposed flow showed a significant improvement in terms of scalability and performance and can be integrated and used in the advancement of the applications such as smart devices like voice assistant enabled speakers, offline voice typing in mobile devices, speech summarization and speech context moderation etc. However, there is always a room for improvement, some future research can still be performed on improving the inference of the proposed method in the constrained systems since faster inference with least resources is always the call of state-of-the-art methods. Exploring and researching to add novelties such as speech content moderation in ASR systems seems one of the promising directions of future work.

References

Arun Babu, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, et al. 2021. Xls-r: Self-supervised cross-lingual speech representation learning at scale. *arXiv preprint arXiv:2111.09296*.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2018. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. *arXiv preprint arXiv:1809.01431*.

Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–197.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.

Samuel Kriman, Stanislav Beliaev, Boris Ginsburg, Jocelyn Huang, Oleksii Kuchaiev, Vitaly Lavrukhin, Ryan Leary, Jason Li, and Yang Zhang. 2020. Quartznet: Deep automatic speech recognition with 1d time-channel separable convolutions. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6124–6128. IEEE.

Somshubra Majumdar, Jagadeesh Balam, Oleksii Hrinchuk, Vitaly Lavrukhin, Vahid Noroozi, and Boris Ginsburg. 2021. Citrinet: Closing the gap between non-autoregressive and autoregressive end-to-end models for automatic speech recognition. *arXiv preprint arXiv:2104.01721*.

Anna Mauranen and Svetlana Vetchinnikova. 2017. Chunks in which we process speech. In *14th International Cognitive Linguistics Conference (ICLC), Tartu, July*, pages 10–14.

Edwin G Ng, Chung-Cheng Chiu, Yu Zhang, and William Chan. 2021. Pushing the limits of non-autoregressive speech recognition. *arXiv preprint arXiv:2104.03416*.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Samuel Thomas, Michael L Seltzer, Kenneth Church, and Hynek Hermansky. 2013. Deep neural network features and semi-supervised training for low resource speech recognition. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6704–6708. IEEE.