Comparative study between addition of one MAMBA block to Wav2Vec2 Pretrained model and Vanilla Pretrained model

Stanford CS224N Custom Project

Puchiss Panitpotjaman
SCPD Student
Stanford University
puchiss@stanford.edu

Abstract

Speech-to-text NLP allows us to quickly transcribe the sentence effortlessly. The application for transcribing speech has been widely used in numerous applications such as voice controlled robotics [7], AR voice command [8], Alexa (Amazon home assistant) [9] and creating transcript summary of long meeting conversation. With more precised model implementation, speech to text NLP will allow human to communicate with the computer or machine more effectively. The state of the art performance of Speech to Text is Wav2Vec2 model [6] with only 8 percent error rate in WER metrics. Therefore, wav2vec2 will be used to investigate how we can improve the performance of Wav2Vec2 through finetuning with TIMIT dataset [5]. In this project, we propose that addition of one MAMBA layer [4] will help increase the performance of the wav2vec2. We have reimplemented MAMBA to make the code more easier to follow. Then, we integrate MAMBA inside wav2vec2 by inheriting the wav2vec2forCTC object and add MAMBA layer inside. The result has shown that the hypothesis is true. The WER of vanilla wav2vec2 is 28.4 while the custom wav2vec2 (MAMBA addition) has the error rate of 27.2. This implies that addition of MAMBA layer can create more efficient speech to text model than wav2vec2. This finding emphasizes that addition of one MAMBA layer helps improving the performance of wav2vec2. Further implementation of MAMBA on other pretrained model may help us to validate our findings. More evaluation metrics are needed to observe the efficiency of this custom wav2vec2 model.

1 Key Information to include

- Mentor: Shijia Yang

2 Introduction

Audio speech recognition (ASR) software facilitates human from manually annotate and create the transcription for long conversation. Moreover, the software can be used to help machine understand word by word for further integration in various applications. Voice command robots [7] are one of the application of this software. Without further need to use console or computer command, human can command the action seamlessly through voice similar to talking to human. AR voice command [6] is another interesting application since it can help human to command the function without having to move their hand. Voice command can help remove the needs for hardware object to input command making system to be more versatile. These applications of speech to text can further boost adoption of AR due to its simplicity. This project aims to boost the performance of wav2vec2 model (state of the art model for ASR). The proposal of MAMBA to be replacement for transformer has been widely speculated. As known, transformer based NLP model has dominated the field for 7 years [10]. Most of the pretrained model are transformer based. Customize the full model may require extreme engineering and advance level programming skills. Therefore, I have proposed an extra integration of one MAMBA layer to observe the effect on training performance and WER metrics of wav2vec2. This finding may inspire others to do the same thing to other pretrained model. After implementation, our method requires only two extra lines inside the code which is simple enough for other people to adopt.

3 Related Work: Wav2Vec2 Model

The Wav2Vec2 paper proposes a breakthrough of the architecture from wav2vec with state of the art performance of 1.8/3.3 WER on clean Librispeech dataset [6]. Nevertheless, the model demonstrates that after training for 1 hour of audio file, it can outperform other model which have been trained with 100 hours of audio file [6]. This level of performance has been acheived through usage of self-supervised learning from raw audio file. The first step is to encode the audio file to create latent representation which will be used by transformer to create context representation. Then, the contrastive loss is calculated from the output of encoder and context representation on unlabeled speech. These processes are in pre-training step. The goal of this step is to make the neural network able to find correct quantized latent audio representation from set of distractors.

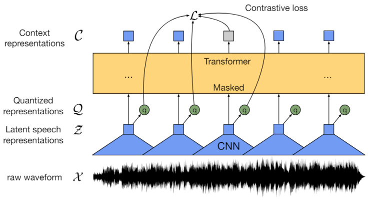


Figure 1: The structure of Wav2Vec2 Model [6]

The next step is to finetune the model with labeled data through Connectionist Temporal Classification (CTC) loss. In the finetuning step, one linear projection is made on top of the context representation layer (output of the transformer) to output the class probability of vocabulary. Now, we can find the CTC loss from these output and finetuning the model. Wav2Vec2 has been tested with ultra-low resources audio file and able to outperform other models. Wav2Vec2 model has return 4.8/8.2 WER on clean Librispeech dataset [6]. Therefore, this model is the best choice for finetuning audio speech in this project. The implementation of Wav2Vec2 are freely avialable in website [3]. We can easily create an adaptation version of Wav2Vec2 and able to customize the model to be in our own implementation.

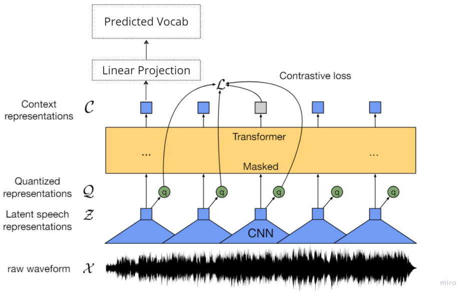


Figure 2: The structure of Wav2Vec2 Model in finetuning step [6]

4 Related Work: MAMBA Architecture

MAMBA [2] utilizes the concept of state space mechanism instead of attention mechanism. State space mechanism has been proposed through many research papers yet it can not reach the same capabilities of transformer in many metrics [2]. What makes MAMBA stands out is the proposition of S6 algorithm which consists of state space model (s4) and selective scan algorithm inside to create MAMBA [2]. To understand the state space model, we need to deep dive into discretization of state space model.

$$h'(t) = \boldsymbol{A}h(t) + \boldsymbol{B}x(t) \qquad (1a) \qquad\qquad h_t = \overline{\boldsymbol{A}}h_{t-1} + \overline{\boldsymbol{B}}x_t \qquad (2a)$$

$$y(t) = \boldsymbol{C}h(t) \qquad (1b) \qquad\qquad y_t = \boldsymbol{C}h_t \qquad (2b)$$

Figure 3: The discretization of state space model [2]

The equation (1a) and (1b) represents the continuous time step of state space model. However, we would like to discretize to work on our model. The equation (2a) and (2b) represents a discrete time step of the state space model. h_t will be the inverse C matrix times to y_t. Now, we are left with finding appropriate $\overline{A}$, $\overline{B}$, C matrix. The paper proposes the equation to find the $\overline{A}$, $\overline{B}$ through equation shown below.

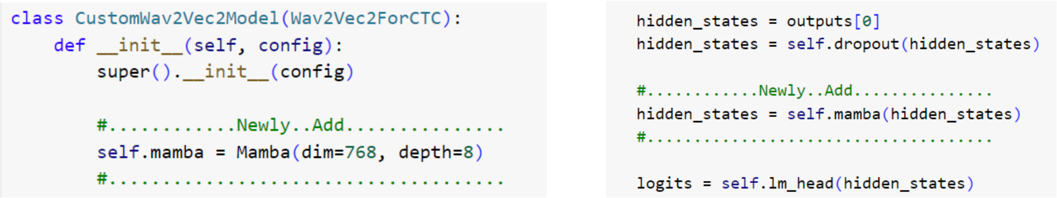
$$\overline{\boldsymbol{A}} = \exp(\Delta\boldsymbol{A}) \qquad \overline{\boldsymbol{B}} = (\Delta\boldsymbol{A})^{-1}(\exp(\Delta\boldsymbol{A}) - \boldsymbol{I}) \cdot \Delta\boldsymbol{B}$$

Figure 4: $\overline{A}$, $\overline{B}$ equation [2]

We can find Δ, A, B, C through linear projection of x which are learnable parameters. The code implementation in this project is based on the original code provided by Albert Gu and Tri Dao [4].

5 Approach

The approach starts at creating the code that is working with vanilla Wav2Vec2forCTC model. We choose Wav2Vec2forCTC model because the model is using in finetuning task. We can use the existing google colab that have already working in finetuning task of the timit dataset [5] from [1]. After we have a working code for this task, we are seeking to customize the Wav2Vec2forCTC model. First, we take a look at the raw source code of Wav2Vec2forCTC model [3]. We can just inherit the Wav2Vec2forCTC class from transformers library [3]. Then, we need to overwrite the forward function and add one mamba layer as attribute inside the Wav2Vec2forCTC [3]. The picture below show the two additions of line inside the class (forward function and init).



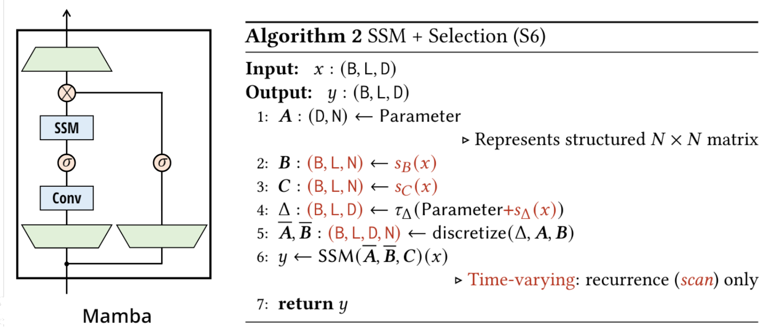
```
class CustomWav2Vec2Model(Wav2Vec2ForCTC):
    def __init__(self, config):
        super().__init__(config)

        #...........Newly..Add..............
        self.mamba = Mamba(dim=768, depth=8)
        #..................................
```

```
hidden_states = outputs[0]
hidden_states = self.dropout(hidden_states)

#...........Newly..Add..............
hidden_states = self.mamba(hidden_states)
#..................................

logits = self.lm_head(hidden_states)
```

Figure 5: Adding MAMBA attribute to the model and changing the forward function inside the model

The next step is to create a MAMBA object. The first step is to create a model of MAMBA architecture [2] from the figure in the left to recreate a MAMBA block. Then, the state space model with selective scan function can be created by following the pseudocode from MAMBA paper itself [2].

Now, the new architecture is ready to be used for finetuning task. The new architecture of CustomWav2Vec2forCTC can be shown in the diagram below. In the finetuning task, we will use TIMIT dataset from Linguistic Data Consortium which can be downloaded from hugging face. The downloaded data need to go through preprocessing methods in order to be able to efficiently compatible with the model and trainer from hugging face library.



```
Algorithm 2 SSM + Selection (S6)
Input: $x$ : (B, L, D)
Output: $y$ : (B, L, D)
    $A$ : (D, N) ← Parameter
                                  ▷ Represents structured $N \times N$ matrix
    $B$ : (B, L, N) ← $s_B(x)$
    $C$ : (B, L, N) ← $s_C(x)$
    $\Delta$ : (B, L, D) ← $\tau_\Delta$(Parameter+$s_\Delta(x)$)
    $\overline{A}, \overline{B}$ : (B, L, D, N) ← discretize($\Delta, A, B$)
    $y$ ← SSM($\overline{A}, \overline{B}, C$)($x$)
                                  ▷ Time-varying: recurrence (scan) only
    return $y$
```

Figure 6: Architecture of MAMBA and pseudocode of Selective Scan SSM [2]

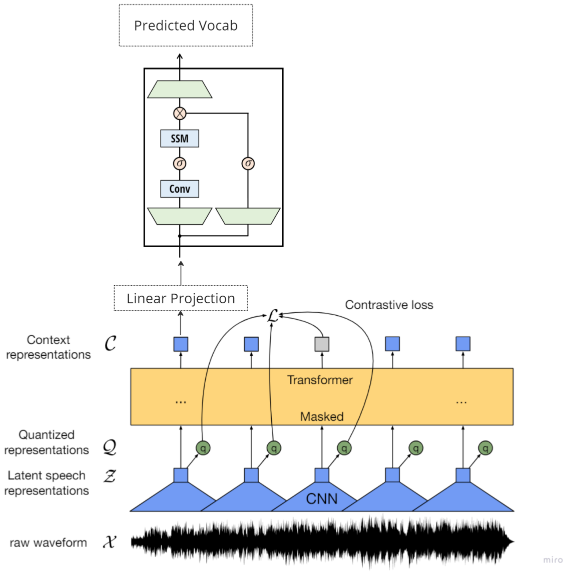


Figure 7: The architecture after integration of MAMBA to Wav2Vec2forCTC [2], [6]

6 Experiments

6.1 Data

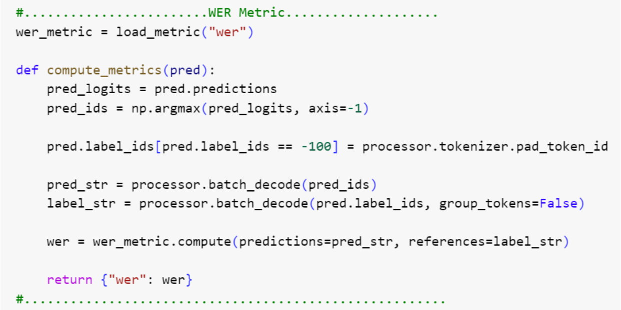
The datasets used to finetune the model are downloaded from hugging face hub. The dataset comes from TIMIT, Linguistic Data Consortium [5] which consists of 4620 files of maximum 4 seconds of human voice file including the data related to the owner of the voice such as dialect region, phonetic details and word for training. However, in this task, they are not important and will be omitted. Another 1680 files of maximum 4 seconds of human voice file are used in testing set and evaluation.

6.2 Evaluation method

We will use WER metric to measure the error rate of the model in predicting the sound. 0.3 WER means that 30 percent of the predicted words are incorrect. The evaluation function can be described through the function shown below. The function has already been created from the original notebook [1]. Therefore, we do not have to create our own function.

6.3 Experimental details

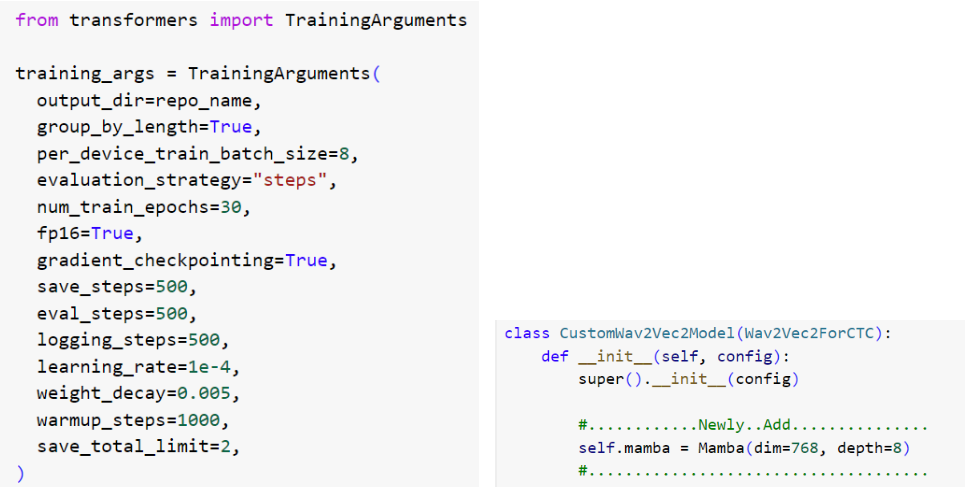
The model configuration in Wav2Vec2 are following the original notebook of wav2vec2 [1] which can be shown in the picture below. In the MAMBA layer, we have investigated the shape and size



```
#........................WER Metric....................
wer_metric = load_metric("wer")

def compute_metrics(pred):
    pred_logits = pred.predictions
    pred_ids = np.argmax(pred_logits, axis=-1)

    pred.label_ids[pred.label_ids == -100] = processor.tokenizer.pad_token_id

    pred_str = processor.batch_decode(pred_ids)
    label_str = processor.batch_decode(pred.label_ids, group_tokens=False)

    wer = wer_metric.compute(predictions=pred_str, references=label_str)

    return {"wer": wer}
#.......................................................
```

Figure 8: WER metric defined in the original notebook [1]

of the output from wav2vec2 and adapt the layer accordingly. We found that the linear projection of the last layer in wav2vec2 has the shape of (8 (batch size), vary (the dimension is not fixed), 768 (dimension after linear projection)). Therefore, we set the dimension of MAMBA layer to have the depth 8 and dimension of 768.



```
from transformers import TrainingArguments

training_args = TrainingArguments(
  output_dir=repo_name,
  group_by_length=True,
  per_device_train_batch_size=8,
  evaluation_strategy="steps",
  num_train_epochs=30,
  fp16=True,
  gradient_checkpointing=True,
  save_steps=500,
  eval_steps=500,
  logging_steps=500,
  learning_rate=1e-4,
  weight_decay=0.005,
  warmup_steps=1000,
  save_total_limit=2,
)
```

```
class CustomWav2Vec2Model(Wav2Vec2ForCTC):
    def __init__(self, config):
        super().__init__(config)

        #...........Newly..Add..............
        self.mamba = Mamba(dim=768, depth=8)
        #...................................
```

Figure 9: Model configuration [1] and the MAMBA layer setting

6.4 Results

The main findings is that the addition of MAMBA layer results in lower WER in the model than the Vanilla. The WER of Vanilla model is 0.284 or 28.4 percent of incorrect words prediction while the WER of Custom model is 0.272 or 27.2 percent of incorrect words prediction. While training, I have also found that the loss clearly show distinction in WER. Nevertheless, another critical factor is training time. Addition of MAMBA increase training time from 1.5 hours to 6.5 hours. This is the disadvantage of the model since increasing the layer of MAMBA layer also increases time for training by 4.3 times. The result from this model is as expected meaning that addition of MAMBA layer truly helps the system to be more efficient, converge faster in term of epochs but slower in term of time usage. The image below can illustrate the progress and differences.

WER Metrics Comparison on test set	
Models	WER Metrics
Vanilla Wav2Vec2	0.284
Custom Wav2Vec2	**0.272**

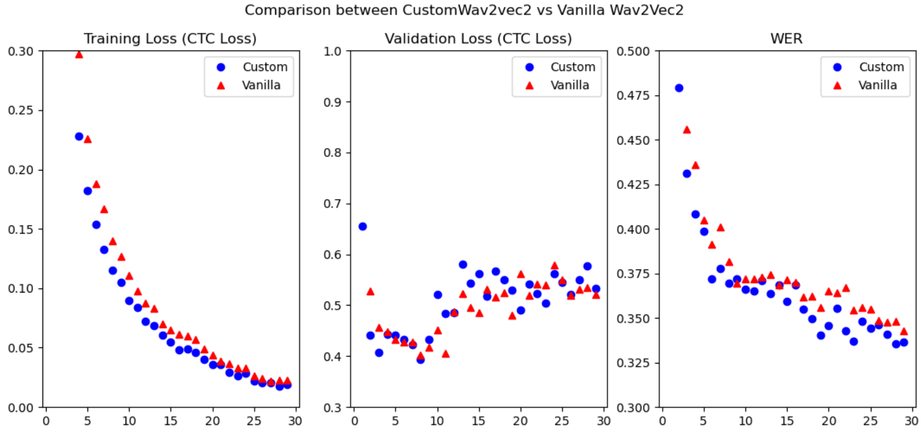


Figure 10: Comparison of Custom and Vanilla Wav2vec2 during training, validation and WER Metrics

7 Analysis

The working principle of the wav2vec2 model is transcribing sound wave to letter that responsible for the sound. An example of predicted letter is here. This is the sentence that I spoke to the model "My name is John. I love jellyfish". The output of the model is "[PAD] [PAD] [PAD] m m m y y [PAD] [PAD] | n n a a a m e e e e [PAD] [PAD] | | i s s s | [PAD] j j j j o o h h h h h n n n | [PAD] [PAD] | [PAD] [PAD] i i i [PAD] [PAD] | l l l u u v v e e e e [PAD] | j j j e e l l [PAD] [PAD] l l l y y | | [PAD] f f f i i s s s s s [PAD] h h h [PAD] [PAD] [PAD] [PAD] | | | [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]". Then, we can combine those characters together to form a word through CTC. CTC will group the same token together to form a word. The resulting sentence is "my name is John i luve jelly fish". The word love is incorrect since it transcipts to luve. Notice that the recurrence of l in jelly fish can be distinguished by having "[PAD]" token between character. "|" token will split the word with space so that the system know that it is two differences word. The same principle goes with CustomWav2Vec2 model.

8 Conclusion

The speech to text model can be implemented through Wav2Vec2 model. We have implemented a custom model which performs better than Wav2Vec2 model in term of WER Metric (0.284 vs 0.272). However, the drawback of custom Wav2Vec2 is that it takes 6.5 hours to train compared to 1.5 hours in Vanilla. I believe that the long time to train is due to engineering problem which can be optimised and improved. Moreover, to make a stronger claim, we need to use mamba block in other task as well to validate that addition of MAMBA block truly help the model to perform better. Other error metrics should also be included and used to evaluate the effectiveness of the model.

9 Ethics Statement

The ethical challenge of this project is that we can use it to transcript the conversation wrongly and result in misinformation. If it is used in judicial setting, it can lead to serious misintepretation. Moreover, the user can manipulate the transciption by hand and blame the misinformation to AI system to ignore the responsibility in intended misinformation. Another ethical concern is that it can violate the privacy of the user. Hacker can implant this AI system to the hardware and extracting the heavy audio file to text file which can be easily sent to hacker. We can mitigate the ethical violation by adding permission file for AI and saving some copy of the file that we collect so that we can know the true version of the file transcipted by the AI system. However, we should make the file to have expiration date in accordance with the law.

References

[1] Platen, P. von. (2021, March 12). Fine-tune WAV2VEC2 for English ASR in hugging face with transformers. https://huggingface.co/blog/fine-tune-wav2vec2-english

[2] Gu, A. and Dao, T. (2024, May 31) Mamba: Linear-time sequence modeling with selective state spaces, arXiv.org. Available at: https://arxiv.org/abs/2312.00752v2

[3] Hugging Face Transformers - transformers 4.5.0.dev0 documentation. Available at: https://huggingface.co/transformers/v4.5.1/index.html

[4] Gu, A. and Dao, T. (2023) State-spaces/mamba: Mamba SSM architecture, GitHub. Available at: https://github.com/state-spaces/mamba

[5] Garofolo, John S., et al. TIMIT Acoustic-Phonetic Continuous Speech Corpus LDC93S1. Web Download. Philadelphia: Linguistic Data Consortium, 1993.

[6] Baevski, A., Zhou, H., Mohamed, A., Auli, M. (2020, October 22). WAV2VEC 2.0: A framework for self-supervised learning of speech representations. arXiv.org. https://arxiv.org/abs/2006.11477

[7] Naeem, B., Kareem, W., Saeed-Ul-Hassan et al. Voice controlled humanoid robot. Int J Intell Robot Appl 8, 61–75 (2024). https://doi.org/10.1007/s41315-023-00304-z

[8] Sheldon, Aron Dobbs, Tiara Fabbri, Alessandra Gardner, Nicole Haeusler, M. Hank Ramos, Cristina Zavoleas, Yannis. (2019). PUTTING THE AR IN (AR)CHITECTURE: Integrating voice recognition and gesture control for Augmented Reality interaction to enhance design practice. 10.52842/conf.caadria.2019.1.475.

[9] Amazon Alexa. https://www.alexa.com/

[10] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I. (2023, August 2). Attention is all you need. arXiv.org. https://arxiv.org/abs/1706.03762