Language Processing Phonetic-to-word After Speech Recognition (P2W)

B08902017, 游一心
B08902051, 施宇飛
B08902093, 石諾盟
B08902116, 賴世宗
B08902134, 曾揚哲

June 23, 2021

Contents

1 Abstract

The Natural Language Processing has been a major field of study in Machine Learning for several decades. In this paper, the main focus is on Language processing Phonetic-to-word after Speech Recognition. Our goal is to turn the input, which had been processed with Kaldi toolkit Acoustic Model to probability distributions of phones, into Bert contextual word-embeddings, via Transformer and Conformer models.

After the experiments, we will evaluate the performance and explain the result, also there will be some improvement suggestions following. At the end of the paper, we will talk about the applications and prospects of phone to word embedding technique.

2 Introduction

The field of Natural Language Processing has evolved to the point that it serves as a bridge for machines to "understand" whatever audio, preferably human-speech, fed to it. Such advancements in the field leads to so many interesting prospects for the future. This is a future where machines can process and properly understand in just an extremely short time more than any human being can understand in a lifetime, improving the rate at which the data is being collected as well as its quality . It's also a future, where people can have a properly coherent and responsive conversation with assistants better than the likes of SIRI, Alexa or Cortana. But last but not least, this future where machines can express themselves with a hint of humanity to it, accomplished by a pitch perfect recreation of the intricacies of the human voice. The good news is with the growth rate of the Natural Language Processing field, such progresses are near.

3 Dataset

The audio data are collected from MATBN(中文廣播新聞語料庫), which collects audio data from news of Public Television Service and labels the corresponding text by humans. More details please refer to the link.

Data was first processed with Kaldi toolkit Acoustic Model to probability distributions of phones, in this part, it has already been preprocessed by TA.

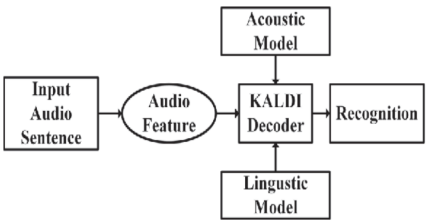

Figure 1: Kaldi's acoustic model MATBN as input

In our experiment, we divide data into train, validation and test data with a ratio 9:1:1 to select model and evaluate performance of models.

4 Extant Methods

4.1 Feature Extraction

4.1.1 Definition

As human beings, we are gifted with the ability to take action with respect to our surrounding audio stimuli. However, we have to convert the recorded sound to specific form so that the computing devices comprehend what they received. The sound we hear is constituted by multiple factors rather than by a well-structured form like within our computers or mobile phones.[5] However, thanks to numerous researchers having already devoted in sound analysis, we have developed several methods for sound feature extraction. Below, we introduce some algorithms widely adopted by extant speech recognition models.

4.1.2 MFCC

Mel **F**requency **C**epstral **C**oefficient, abbreviated MFCC, is already used in multiple purposes such as identifying airline reservation and current voice recognition. Its features are extracted by filtering speech signals linearly at low frequency to high frequency logarithmically. [5] Each feature contains tones of varying frequency whose pitch is computed on the **Mel scale**. The common computation procedure could be depicted as the list below.

1. The speech signals are split into frames by windowing.
2. Apply FFT to find the power spectrum of each frame.
3. Proceed **filterbank** on the power spectrums found in step 2, using mel-scale.
4. Take logarithm of the translation and apply DCT onto it.

4.2 Model Architecture

There are various models adopted by previous work in speech recognition. For instance, Feed-forward neural networks were explored 20 years ago. Recently, DNNs have been introduced to the ASR pipeline in some state-of-the-art research. Convolution are also employed in some speech work, too. RNN, typically LSTM, have been shown working well with convolution layers, while both bi-direction (Bi-LSTM) and unidirectional ones have been explored. Other than the aforementioned models, some other architectures stand on shoulder of giants, as language model pre-training has been shown to be effective improving many NLP tasks.[7] Here, we introduce some prominent work in ASR realm.

1. **wav2vec 2.0**[6]: Semi-supervised learning
   convolution encoder + layer norm + GELU + Transformer + Gumbel softmax
2. **BERT**[7]: pre-training + fine-tuning
   multilayer bidirectional Transformer encoder and decoder, bidirectional self-attention
3. **Kaldi Toolkit**[8]:
   LSTM + Recurrent Dropout + BatchNorm + Monophone Regularization
4. **Pushing the Limits of SSL for ASR**[9]: iterative self training + pre-training
   Conformer + pre-training wav2vec 2.0 +
   noise student training[10] + adaptive SpecAugment [11]

5 Our Method

5.1 Differences from extant methods

We extract features from audio file by Kaldi Acoustic Model and get phone distributions, not MFCC traditionally. Moreover, we wish to generate contextual word embedding instead of words since if there are some errors among models, using word embeddings can still preserve some part of the meaning of a word and therefore reduce the propagation of uncertainty. However, if we use words as output, while generating a wrong word, it will cause a huge impact on the model of downstream task.

5.2 Pipeline Diagram

Below is our pipeline diagram. First, we turn sound files into phone distributions by Kaldi Acoustic model. Second, we use our models to turn those distributions into Bert's contextual word embeddings(obtained from the hidden state of last layer of a pretrained Bert model. Finally, we wish those word embeddings can be used in downstream NLP task to improve performance such as NLU, DST and so on.

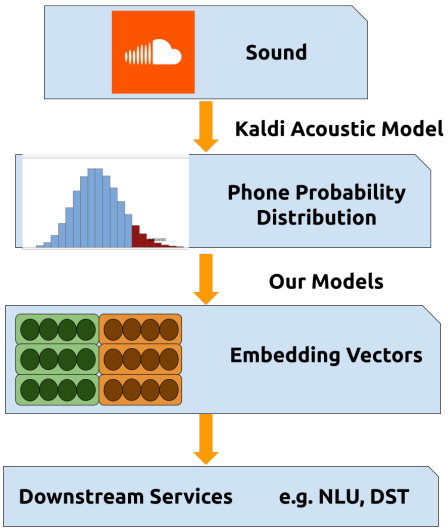

Figure 2: Flowchart

5.3 Model Structure

1. **Sequence to Sequence**

   Sequence to sequence (stylished as Seq2seq) is a model architecture widely used when it is needed to handle different length of input sequence and output sequence and is applied on lots of task like machine translation, summarization and so on. Generally, a Seq2seq model includes an encoder and a decoder. Encoder is in charge of extracting the information of input sequence while decoder take those input to generate output sequence.

2. **Model**

   (a) **Transformer**

We use transformer [12], as our basic model structure as the below picture shown.

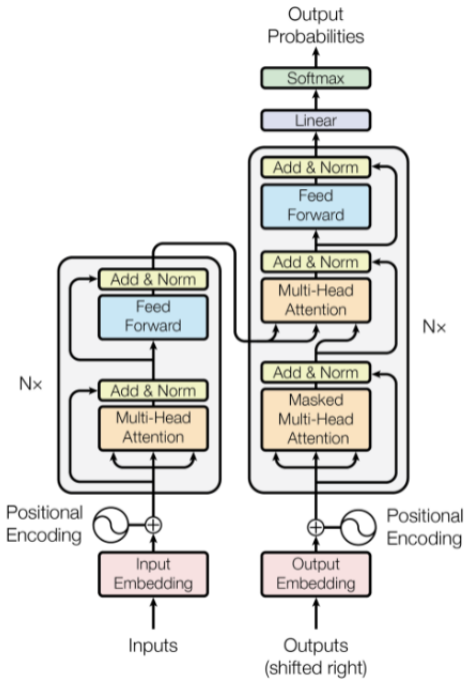

Figure 3: Structure of transformer

(b) **Conformer**

To improve performance, we replace the encoder with that of another model, conformer [13], as our another model structure in experiments. The structure of encoder is as the picture shown below.

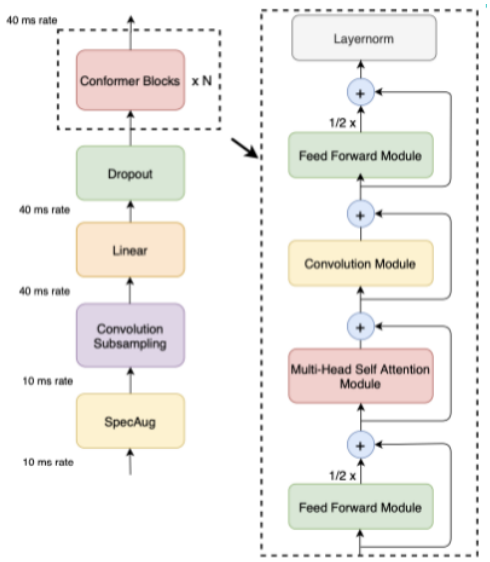

Figure 4: Structure of conformer encoder

3. Methods used to improve performance in experiments

(a) **Nucleus sampling to generate input sequence**

To prevent our model from learning directly from non-deterministic information, such as the probability distribution of phones, in some of our experiments, we first generate a sequence of phones by nucleus sampling with top_p = 0.5 before input into our model. More details about nucleus sampling, please refer to Appendix 8.1.

(b) **Schedule sampling**

We implement schedule sampling method mentioned in [14] to address the exposure bias between training and testing phase.

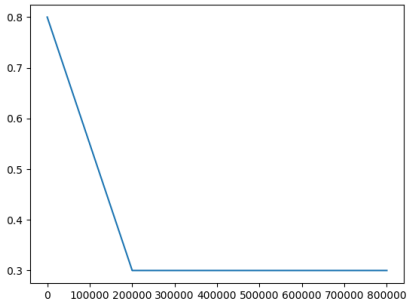

Figure 5: Probability to use labels while training vs. steps

The picture above shows the probability that our decoder use labels as input linearly decay from 0.8 to 0.3 when training steps increase.
More details about schedule sampling, please refer to Appendix 8.2.

(c) **Gradient Accumulation**

Due to the lack of training facilities (by the enormous size of data), we could only use batch size as 1 so that the gradient accumulation is needed. We set the gradient accumulation to 32 for implementing 32 batch size like.

$$\text{accumulated} = \sum_{i=0}^{N=32} grad_i$$

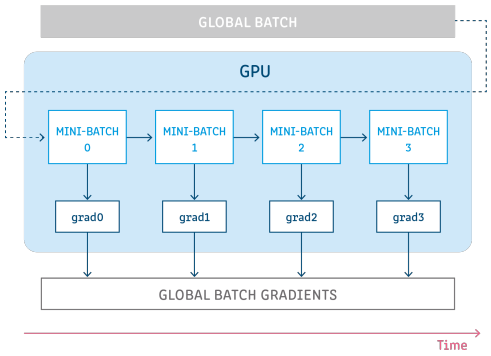

Figure 6: gradient accumulation

5.4 Experiments

We use AdamW with $lr = 10^{-5}$ and cosine warm-up scheduler as optimizer on all experiments and test four set of methods in experiments.

1. Use transformer encoder and decoder and input phone distribution.
2. Use transformer encoder and decoder and input phone sequences after nucleus sampling.
3. Use conformer encoder and transformer decoder and input phone distribution.
4. Use conformer encoder and transformer decoder and input phone sequences after nucleus sampling.

5.5 Evaluation Method

We evaluate the result of experiments by calculating the cosine similarity between the word embedding sequence generated by our model and label provided by TA. More detail about cosine similarity, please refer to Appendix 8.3.

5.6 Performance

Below is the performance of our experiments.

model	nucleus sampling	schedule sampling	cosine similarity
Transformer	No	Yes	0.4514
Transformer	Yes	Yes	0.4395
Conformer	No	Yes	0.5356
Conformer	Yes	Yes	0.5846

6 Future Work

1. Improving Evaluate Method: Could use denoise auto-encoder.

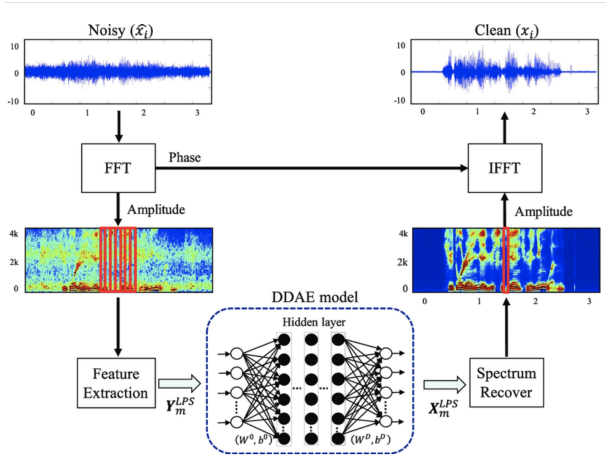

Figure 7: Structure of a deep denoising auto-encoder (DDAE)-based noise reduction (NR) system. FFT, fast Fourier transform; IFFT, inverse fast Fourier transform.

2. We this this task is a generative task rather than a regressive task, so we can try some generative method like using MelGAN to train the discriminator in first part

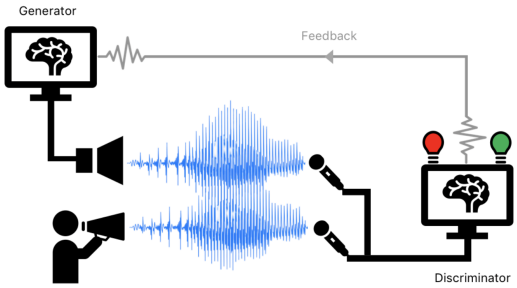

Figure 8: MelGAN

3. Using larger model structure.
4. Train longer on larger training dataset or improve validation set approach.
5. Try other generation method to generate our input such as beam search, top_k sampling or changing the temperature.

7 Conclusion

Due to the time limitation of the project, our performance is not good enough to put in application. Despite the fact, our effort in this project proves that this method is effective and some methods proposed like nucleus sampling and conformer improve performance significantly. If we train longer on a larger dataset, performance will definitely rises to a level that can be used in downstream task.

8 Appendix

8.1 Nucleus Sampling

Before digging into nucleus sampling, we first need to learn about top_k and top_p filtering. Top_k filtering probability by keeping only k tokens with the highest probability, while top_p filtering probability by keeping tokens until aggregate probability ≥ P. Difference type of distribution suits difference filtering.

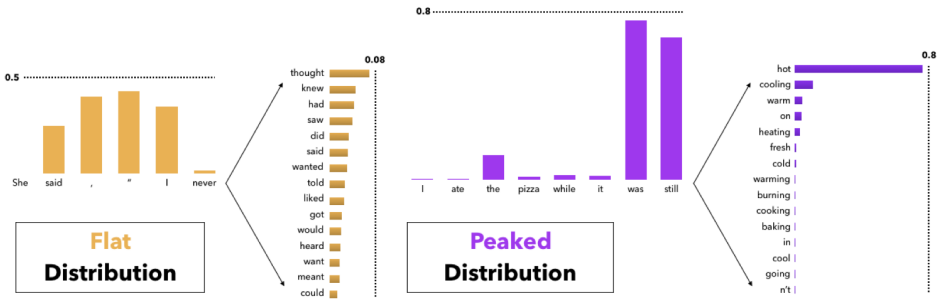

Figure 9: Types of distribution

Top_k and Top_p(a.k.a Nucleus) sampling is a method that sample data from a distribution after top_k and top_p filtering.

8.2 Schedule Sampling for transformer

Schedule sampling is a method proposed to solve the exposure bias between training, testing phase and is usually used in traditional LSTM-based model and is hard to implement on transformer-based model. In [14], it proposes to pass decoder twice during training phase in order to achieve schedule sampling while the first time use labels as input and the second time uses output of decoder at first time mixed by labels as input. Notice that only back-propagation at the second iteration results in way better performance and we use this method in our experiments as well.

8.3 Cosine Similarity

Since we have to measure the similarity of two embedding vector, cosine similarity will find the dot product of two vector divided by their magnitude.

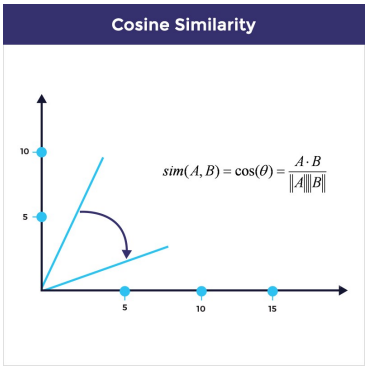

Figure 10: cosine similarity of two vectors

8.4 Our code

https://github.com/Joshuaoneheart/Fintech-Final-Report

8.5 Records of meetings

https://hackmd.io/8nDChxcyQqGNLUIw7gSoFQ

8.6 Work Division

游一心：書面報告、實驗設計、程式撰寫、進行實驗、口頭報告
施宇飛：書面報告、口頭報告、ppt 製作、進行實驗
石諾盟：書面報告、進行實驗、ppt 製作
賴世宗：書面報告、進行實驗、ppt 製作、會議記錄
曾揚哲：書面報告、進行實驗、ppt 製作、會議記錄、程式撰寫

References

[1] Kaldi toolkit Acoustic Model
https://kaldi-asr.org/doc/model.html

[2] THE CURIOUS CASE OF NEURAL TEXT DeGENERATION
https://arxiv.org/pdf/1904.09751.pdf

[3] Ultra Fast Audio Synthesis with MelGAN
https://blog.descript.com/ultra-fast-audio-synthesis-with-melgan/

[4] Figures 3,. Structure of a deep denoising autoencoder(DDAE)
https://www.researchgate.net/figure/Structure-of-a-deep-denoising-autoencoder-DDAE-based-noise-reduction-NR-system-FFT_fig1_322672182

[5] Alim, S.A., Rashid, N.K. (2018). Some Commonly Used Speech Feature Extraction Algorithms.

[6] A. Baevski, H. Zhou, A. Mohamed, M. Auli. (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations

[7] J. Devlin, M. Chang, K. Lee, and K. Toutanova. (2018) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

[8] M. Ravanelli, T. Parcollet, Y. Bengio. (2019). The PyTorch-Kaldi Speech Recognition Toolkit

[9] Y. Zhang, J. Qin, D. S. Park, W. Han, C.-C. Chiu, R. Pang, Q. V. Le, and Y. Wu, (2020). "Pushing the limits of semi-supervised learning for automatic speech recognition,"

[10] Q. Xie, M.T. Luong, E. Hovy, and Q. V Le. (2020) Self-training with noisy student improves imagenet classification

[11] D. S. Park, W.Chan, Y. Zhang, C.C. Chiu, B. Zoph, E.D. Cubuk, and Q. V.Le. (2019) Specaugment: A simple data augmentation method for automatic speech recognition.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin (2017) Attention Is All You Need

[13] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, Ruoming Pang (2020) Conformer: Convolution-augmented Transformer for Speech Recognition

[14] Tsvetomila Mihaylova, Andre F. T. Martins (2019) Scheduled Sampling for Transformers