End-to-End Speech to Named Entity Recognition System

Xiang Jiang, Teeno Ouyang
Department of Computer Science
Stanford University
xiangj3@stanford.edu, Tianhao_ouyang@outlook.com

Abstract

In this paper, we describe an end-to-end system that takes speech audio as input and output the annotated text with named entities with support of 13 categories. This paper also discussed and compared our model with other existing approaches including traditional Two-Step approach and the existing E2E approach. We've also introduced details of how we applied different techniques such as fine tuning and transfer learning to our model to improve either the efficiency and/or accuracy.

1 Introduction

Automatic Speech Recognition(ASR) and Named Entity Recognition(NER) from text both have been popular deep learning problems and been widely used in different applications. The current two-step approach [2] for Speech to Named Entity Recognition is using the automated transcript by ASR as input to NER. This approach has two major drawbacks: error propagation resulted from the ASR transcript to NER, and information loss such as key-clue features including capitalization and punctuation. We are proposing an End to End system that combines ASR and NER into one pipeline that has higher accuracy and more generalization than popular approach. The pipeline takes the audio tracks as input and outputs the annotated text data with entities. The input audio tracks will be read as spectrogram format and we then uses conv, GRU and FC layers to output the annotated text with entities. Prettified output example in Figure 1:

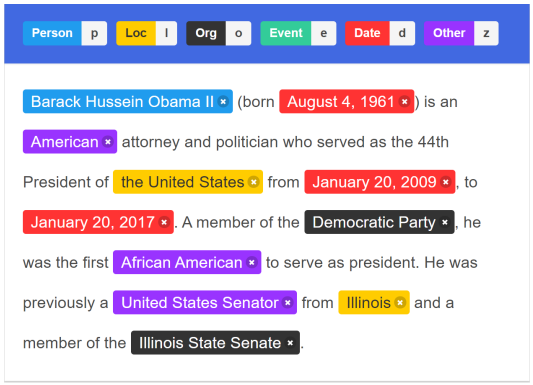

Figure 1: Example Annotation [3]

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related Work

The current existing End to End approach[17] is using two layers of CNN, five layers of Bi-LSTM, and one output layer using FC and softmax as activation. It only supports 3 types of entities, Person, Location, and Organization. We are expanding the number of entities category to 13, including Person, PersonType, Location, Organization, Event, Product, Skill, Address, PhoneNumber, Email, URL, IP, DateTime, and Quantity. Also the neural network model architecture we used are slightly different with existing solution. We've added one more Conv2d layer in the first CNN layers, used Bi-GRU layer instead of LSTM layer, and added Batch Normalization between layers. In addition, unlike the existing approach that uses same "]" character to denote the end of all entities in the text, we've used same character as used in start to denote, that will help reduce the error. For example, Person entity "|Alex|" will have consistent denote both at the start and the end with character 'l'. Comparing to traditional two-step method of Speech Entity Recognition approach, we've brought the same flexibility as two-step method has, also improved the accuracy and efficiency.

3 Dataset & Preprocessing

We have used the initial dataset that contains approximately 1000 hours of English speech in 16KHz without any background noises from LibriSpeech ASR corpus[14]. This audio part of the initial dataset has been firstly passed into a pre-trained ASR model[13], and the output transcript then be fed into a pre-trained NER model[12] to generate an annotated NER dataset. During this process, we have established the two additional filter procedures between ASR output to NER input and NER output to final output. These filter procedures are responsible for filtering low-confidence results and significant anomalies before the data is used for the E2E training. Some manual annotations and fixes are also done. Also the dataset has been divided into three parts, 15% has been used on validation, another 15% has been used on evaluation, and rest has been used for training purposes. Due to the limitation of resources, in both time and computational manner, we've processed 100 hours of audio from the initial dataset and corresponding annotated text with named entities. We've used the the dataset prepossessing[7] program transform the transcript into annotated text with 13 different types of entity representing in 13 different marks[7]. List of the marks in Figure 2:


```
{
    "Person": "|",
    "PersonType": "{",
    "Location": "&",
    "Organization": "~",
    "Event": "!",
    "Product": "(",
    "Skill": "^",
    "Address": "%",
    "PhoneNumber": "#",
    "Email": "@",
    "URL": "/",
    "IP": "*",
    "DateTime": "<",
    "Quantity": ":"
}
```


Figure 2: Named Entity Annotation Marks [3]

4 Methods & Approach

The model that is implemented in this work is based on Baidu DeepSpeech2[1] and the open source Tensorflow implementation[16]. The final model is showing on Figure 3a. The model structure contains a Convolution Network(CNN), a Bidirectional Recurrent Network (RNN), and a Fully Connected Layer(FC) with softmax activation. All the layers in all three network are added extra batch normalization layer to make the training phase faster and more stable. The each Conv2d layers in CNN have 32 filters, with window size (41, 11), stride (2,2), using valid padding. Since it is using valid padding, we've added padding layer before each Conv2d layer. For RNN layers, we've used GRU layers instead of Simple RNN and LSTM, a main reason we used GRU is that GRU provides the gate that would improve the Speech Recognition and also reduce the training time comparing to

LSTM, as reported in [8]. We've used Adam as the optimizer and choose 5e-4 as the initial learning rate. For loss function, because of the natural characteristics of audio, we've used Connectionist temporal classification(CTC)[4] as our loss function[15]. The RNN output gives a distribution of outputs for each input step to CTC and CTC compute the probability of different sequence and marginalizing the over alignments, example flow in figure 3b. By using the ctc loss, we can get around not knowing the alignment between the input audio and the output text. Due to the limitation of resources and dataset, we've applied transfer learning and fine-tuning techniques on this model and separated the training into two steps. First, we train the model using the initial dataset which the output text will only contains 29 types of characters and no annotated named entities. Second, we freeze the CNN and Bi-RNN layers and extend the last FC layer units to support the output of 55 types characters. The extra 26 types contains number 0 to 9 and period, comma, and question mark. In theory, these extra character features can improve model's accuracy by having the network inputs having stronger pattern for some specific entity categories such as IP address, phone number, email, and Email.

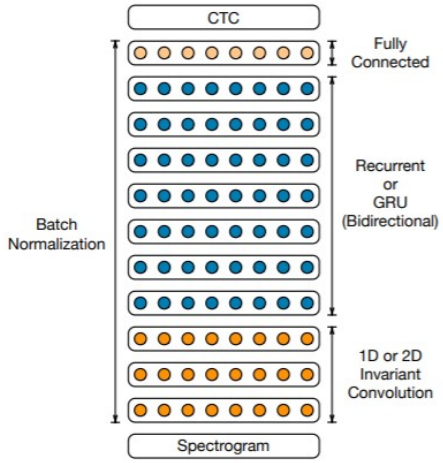

Figure 3a E2E Model Structure [17]

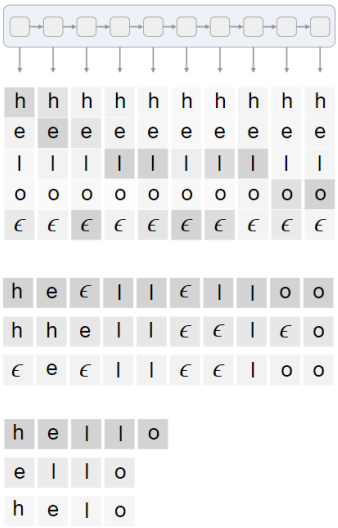

Figure 3b CTC Example Flow[5]

5 Fine Tuning

We've made multiple different approaches before we end up with current model. We've tried different combination of layer types and layer sizes for RNN and added extra layers. Figure 4 showing the previous approaches result we've made.

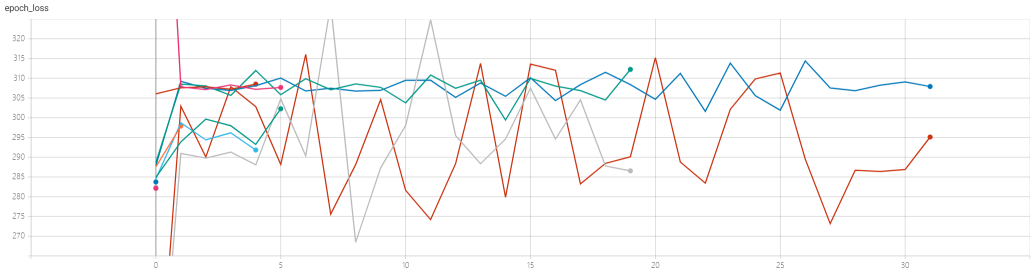

Figure 4 Loss Graph for different approach [17]

Each line in the charts represent a different approach that uses different model structure and/or hyper parameters. We've noticed that during the loss for all these previous model we've tried is either bouncing back and forth or stays relatively constant until we've migrated to the current model architecture. We've firstly trained the model using the initial ASR dataset and discovered that even

though the loss for this model is still not a smooth decrease line due to the mini-batch technique but overall the loss is decreasing. After 16k steps for training, the loss decrease to 0.5. We then applied transfer learning techniques that freeze the CNN and RNN layers and uses our own dataset to train the model. In the loss diagram showing in Figure 5, we've applied the transfer learning techniques on 16k steps(Blue circle).

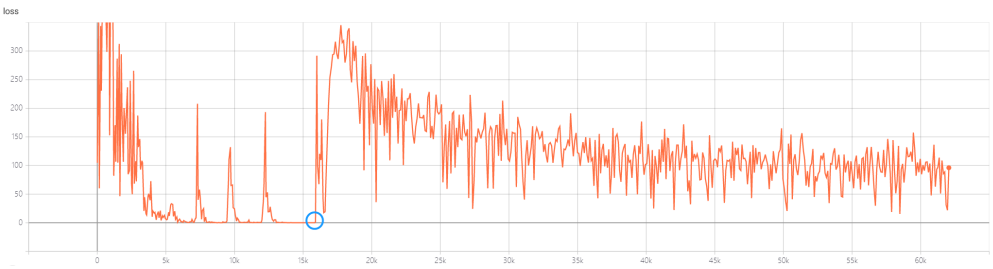

Figure 5 [17]

6 Evaluation

We've used the reserved 15% dataset to evaluate our pipeline. The evaluating dataset has been tested in both traditional two-step implementation and our E2E implementation. The current model has been trained total of 62k steps. Usually in ASR, people Word Error Rate(WER)[11] and Character Error Rate(CER)[9] to evaluate the accuracy of the model, and The slot error rate(SER)[10] is exactly analogous to the word error rate which has been in use as the primary measure of speech recognition performance. We've used Slot Error Rate(SER) as the primary matrix to evaluate our result and used WER and CER as the secondary matrices to help error analysis. Comparing to Two-step and existing E2E approach, we've had significant difference of Slot Error Rate with other approaches showing in Table 1. One great improvement is that our model ends with lowest loss of 0.735 comparing the existing E2E approach which has loss of 5.983.

	WER	CER	SER	Entity Types
Two-step[6]	0.2598	N/A	0.49	7
Existing E2E[17]	0.2469	0.0962	0.16	3
Our Model(before tuning)	1.00	0.9178	1.0	13
Our Model(after tuning)	0.8753	0.7432	1.0	13

Table 1: WER result of different approach [17]

Due to the high Word error rate and high Character Error Rate of hour current model, the Slot Error Rate is also significantly larger than other approaches. In theory, our model should be able to reach WER between 0.21 - 0.23 model and also reach SER 0.49 and lower. Based on the conducted error analysis, we found the main reason for our model has such high SER is our model needs more training steps or epochs in training phase in order to improve WER and CER. We've tried modified the initial learning rate and also adjust RNN layers from Simple RNN to GRU. Also increase the size of initial dataset and tuned other hyperparameter. With aid of the larger dataset and hyperparameter tuning, the system dropped more than 15% CER and WER.

7 Future work

Due to the limitations of time, we've used GRU instead of LSTM as the RNN layer and only applied 1000 hours of Audio as the initial dataset to reduce the training time. Also due to lack of the computational resources, we can only use mini batch size as 16. In the future work, we will increase the mini batch size, increase the dataset size and may add a language model during the training to further improve the WER and CER in order to improve the SER. We will also add data augmentation in the future which we don not have enough time for it this time.

8 Contribution

In here, really appreciate the advise from Teeno on Model structure, dataset preprocessing and ideas of transfer learning and fine tuning.

References

[1] Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Erich Elsen, Jesse Engel, Linxi Fan, Christopher Fougner, Tony Han, Awni Hannun, Billy Jun, Patrick LeGresley, Libby Lin, Sharan Narang, Andrew Ng, Sherjil Ozair, Ryan Prenger, Jonathan Raiman, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Yi Wang, Zhiqian Wang, Chong Wang, Bo Xiao, Dani Yogatama, Jun Zhan, and Zhenyao Zhu. Deep speech 2: End-to-end speech recognition in english and mandarin, 2015.

[2] Antoine Caubrière, Sophie Rosset, Yannick Estève, Antoine Laurent, and Emmanuel Morin. Where are we in named entity recognition from speech? In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4514–4520, Marseille, France, May 2020. European Language Resources Association.

[3] Doccano. Document annotation tool. `http://doccano.herokuapp.com/demo/named-entity-recognition/`.

[4] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 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, 2006. Association for Computing Machinery.

[5] Awni Hannun. Sequence modeling with ctc. *Distill*, 2017. https://distill.pub/2017/ctc.

[6] Mohamed Hatmi, Christine Jacquin, Emmanuel Morin, and Sylvain Meignier. Incorporating named entity recognition into the speech transcription process. In *Incorporating Named Entity Recognition into the Speech Transcription Process*, 08 2013.

[7] Xiang Jiang. Data preprocessing. `https://github.com/xiangj1/E2E`.

[8] Shubham Khandelwal, Benjamin Lecouteux, and Laurent Besacier. COMPARING GRU AND LSTM FOR AUTOMATIC SPEECH RECOGNITION. Research report, LIG, January 2016.

[9] I Scott MacKenzie and R William Soukoreff. A character-level error analysis technique for evaluating text entry methods. In *Proceedings of the second Nordic conference on Human-computer interaction*, pages 243–246, 2002.

[10] John Makhoul, Francis Kubala, Richard Schwartz, and Ralph Weischedel. Performance measures for information extraction. In *In Proceedings of DARPA Broadcast News Workshop*, pages 249–252, 1999.

[11] Iain Mccowan, Darren Moore, John Dines, Daniel Gatica-Perez, Mike Flynn, Pierre Wellner, and Herve Bourlard. On the use of information retrieval measures for speech recognition evaluation. 01 2004.

[12] Microsoft. Azure named entity recognition. `https://docs.microsoft.com/en-us/azure/cognitive-services/text-analytics/how-tos/text-analytics-how-to-entity-linking?tabs=version-3-preview`.

[13] Microsoft. Azure speech to text. `https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/`.

[14] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5206–5210. IEEE, 2015.

[15] Tensorflow. Tensorflow ctc loss. https://www.tensorflow.org/api_docs/python/tf/nn/ctc_loss.

[16] Tensorflow. Tensorflow deep speech. https://github.com/tensorflow/models/tree/master/research/deep_speech.

[17] Hemant Yadav, Sreyan Ghosh, Yi Yu, and Rajiv Ratn Shah. End-to-end named entity recognition from english speech, 2020.