Complete Sentence Detection for Speech Recognition Systems

Machine Learning Project2 Report

Bohan Wang
321293
bohan.wang@epfl.ch

Ke Wang
326760
k.wang@epfl.ch

Siran Li
321825
siran.li@epfl.ch

***Abstract*—In this project, we present a classification model based on state-of-art transformer models to detect the completeness and correctness of a sentence. It is designed specifically for detecting errors in speech recognition systems and takes several typical recognition errors in into account, including false sentence boundary, missing words, repeating words and false word recognition. The classification model can be applied to evaluate the quality of the recognized transcripts, and the optimal model reports over 90.5% accuracy on detecting whether the system completely and correctly recognizes a sentence.**

I. Introduction

It is a easy task for a human being to identify whether a sentence is complete and correct (e.g. ”It is a nice day” is a proper sentence, whereas ”It is a nice” is not). However, such a classification model can be hard for the machines, due to lack of ability to understand the logic of sentences.

A practical application domain for this task is to evaluate the quality of the speech recognition systems. Due to the inability to correctly identify the start or end of sentences, and inability to correctly recognize certain words, the speech recognition system can produce unsatisfying recognitions, which we will detail in Sec. II-C.

Previous researches on detecting erroneous sentences focused mainly on grammatical error detection [1], [2], none of them have focused on dealing with the specific errors emerged in speech recognitions (such as missing words, incorrect sentence boundaries). In addition, previous works on enriching speech recognitions emphasis on finding correct sentence boundaries in whole transcripts [3], [4], a potential problem with their applications is that in real-time speech recognition, we have access to only individual sentences instead of full transcripts, and they don't take other typical speech recognition errors (apart from incorrect sentence boundaries) into account.

Therefore, our project aims to build a classification model that decides whether a recognized sequence of words/phrases is a complete and proper sentence. It is designed based on detecting the typical errors of speech recognitions, to evaluate the recognition quality and provide feedback for possible improvements on the system.

Recently, transformer models have shown state-of-art performance in generating word embeddings and extracting intrinsic features of word sequences. In specific, Bidirectional Encoder Representations from Transformers (BERT) [5], Generative Pre-trained Transformer (GPT) [6] and BIGBIRD [7] have achieved promising performance to learn high quality language representations from large amounts of raw text. The token representations produced by these transformers pre-trained on unsupervised tasks also help improve the performance of a supervised downstream task.

In this project, we fine-tune the pre-trained transformers (BERT, GPT2 and BIG-BIRD) on the speech recognition error detection task, to build a binary classification model detecting speech recognition errors. The performance of sequentially linking BERT embedding and a down-stream text classification network is also studied. We compare and analyze the performances of several classification models. Finally, we ensemble the models through a Random Forest to further improve the performance.

The report is structured as follows: In Sec. II, we describe how the dataset is generated. In Sec. III, we explain the models and experimental design. In Sec. IV, we report the experimental results and discuss them in Sec. V.

II. Data preparation

A. Dataset sources

For the model to have better generalizing capacity, a training set from diverse sources covering diverse topics and occasions is necessary. The following corpus are included in our proposed dataset:

- *News reports* [8]: 143, 000 articles from 15 American publications
- *Ted 2020 Parallel Sentences Corpus* [9]: around 4000 TED Talk transcripts from July 2020
- *Wikipedia corpus* [10]: over 10 million topics
- *Topical-Chat* [11]: human dialog conversations spanning 8 broad topics

From these datasets, we extract sentences and create two datasets for our project.

- **Standard Dataset**: contains 0.3 million sentences from news reports, 0.3 million sentences from Ted corpus, 0.3 million sentences from Wikipedia corpus, 0.2 million sentences from Topical-Chat, in total 1.1 million sentences.

We split the Standard Dataset into train set, ablation set and test set.

- **Large Dataset**: contains 2.3 million sentences from news reports, 0.4 million sentences from Ted corpus, 2 million sentences from Wikipedia corpus, 0.2 million sentences from Topical-Chat; in total 5 million sentences. We split it into train and test set.

We will train and compare performances of various models on the standard dataset. As a comparison, we will test the performance of BERT on the large dataset to see how an enlarged training set affects test performance for this task.

B. Data pre-processing

The following pre-processing steps are applied on the raw dataset:

- **Sentence Tokenization**: split the raw text into sentences
- **Remove sentence ending symbols**: the possible symbols (‘.’, ‘!’, ‘?’) for sentence ending are removed
- **Deal with special tokens**: sentences containing special symbols and non-English tokens (e.g. $'-'$, ϕ, ψ, etc.) are removed. Parentheses and the words inside parentheses are removed (but the other parts of the sentences are kept).
- **Remove short sentences**: the raw data from the corpus contains many short word combinations which are not sentences (e.g. dates, names, headlines). The sentences within 5 words from news reports and Wikipedia corpus are removed. Besides, the sentences within 2 words from Topical-Chat are removed to erase the bias of oral expressions in human dialogues (e.g. names, adjectives),.
- **Remove duplicates**: duplicated sentences are removed.

C. Create positive and negative samples

For creating positive samples, punctuation is removed (except abbreviations such as it’s, Mr., I’ve, etc.) and words are converted to lower case.

For creating negative samples, we try to mimic the errors of the speak recognition system, which are detailed in the following, and we propose corresponding methods to create negative samples with respect to typical errors.

- **False sentence boundary**: When the recognition system fails to find sentence endings and thus fails to correctly separate the sentences, the original sentence would be cut off in the middle and part of the sentence would be assigned to the next sentence (illustrated in Fig. 1 (a)). For such negative samples, we group the sentences by three, and randomly separate the three sentences into 2-4 sentences (so that in average negative samples created in this way would have equal length with positive samples). While choosing random separating points, the genuine sentence separations points, punctuation and typical words for ending sentences (e.g. that, which, because, etc.) are avoided, and thus reduce the probability that a generated sample is still a complete sentence by chance (e.g. ’I like you because you are beautiful’ to ’I like you’.)
- **Missing words**: The system can fail to recognize several words from a sentence, and as a result some words may be missing in the produced transcripts (Fig. 1 (b)). For such negative samples, we randomly remove 2-4 words from a sentence (1 word for sentences within 3 words).
- **Repeating words**: The system can record speakers’ unintended words repeating (Fig. 1 (c)). For such negative samples, we randomly repeat 1-3 words from a sentence (1 word for sentences within 3 words).
- **False word recognition**: The system can make mistakenly recognize word as another word (Fig. 1 (d)). For such negative samples, we randomly replace 1-3 words from a sentence by random words from another sentence (1 word for sentences within 3 words).

Finally, the punctuation is removed and words are converted to lower case.

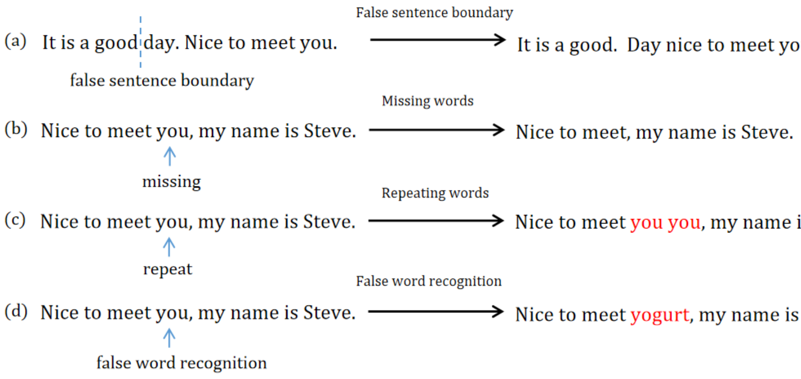
False sentence boundary
(a) It is a good day. Nice to meet you.
It is a good. Day nice to meet yo
false sentence boundary
Missing words
(b) Nice to meet you, my name is Steve.
Nice to meet, my name is Steve.
missing
Repeating words
(c) Nice to meet you, my name is Steve.
Nice to meet you you, my name i
repeat
False word recognition
(d) Nice to meet you, my name is Steve.
Nice to meet yogurt, my name is
false word recognition

Fig. 1: Typical errors in speech recognition system

After creating the positive and negative samples, the sentences longer than 100 words are removed. We create the same number of negative samples as that of positive samples, so that we have a balanced dataset. The ratio between different types of negative samples is 2:1:1:1 (Fig. 2). The reason that the type *False Sentence Boundary* corresponds to two times the number of other negative sample types is that *False Sentence Boundary* contains two types of false sentences, those which are cut off and those which are assigned with extra words.

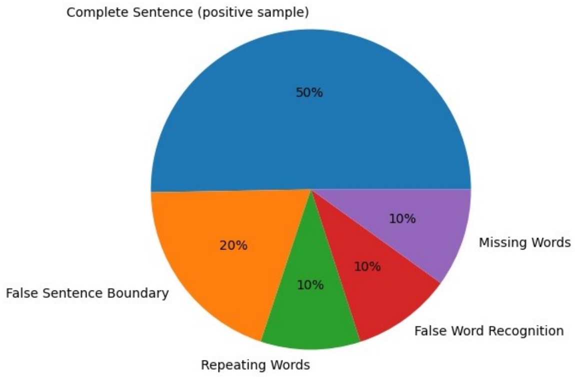

Fig. 2: Ratio for different types of samples.

III. Methodology

In this section, we describe the classification models we use on the task. Three state-of-art transformer models : BERT, GPT2, BIG-BIRD are considered. Besides, we also test the performance of using BERT embedding plus a downstream text classification network. Here the classification networks we consider include bi-direction LSTM and text CNN.

A. Model

1) Transformers: We fine-tune below pre-trained transformers from Hugging face [1].

GPT-2: GPT-2 is a large transformer-based language model with 1.5 billion parameters, trained on a dataset of 8 million web pages to 'predict the next word, given all of the previous words within some text'. [6]. GPT-2 outperforms other language models trained on specific domains (like Wikipedia, news, or books) without using these the datasets [6].

BERT: BERT applies bidirectional self-attention where each token can attend to context to its left and right [5]. Based on this, BERT is pretrained with two unsupervised tasks including predicting the masked tokens and predicting the next sentence [5]. In the project, we use BERT base model (uncased) which is pretrained with the above two objectives on BooksCorpus and English Wikipedia.

BIG-BIRD: BIG-BIRD applies sparse, global, and random attention to process much longer sequences (4096 tokens) [7]. This transformer in Hugging Face is pre-trained on four publicly available datasets: Books, CC-News, Stories and Wikipedia. The pre-trained BIGBIRD can consider various semantic information of the sentence.

2) BERT embedding + classification network: Here we test the performance of using the BERT embedding to embed the words in the sequence, followed by a downstream classification network, either a TextCNN or a BiLSTM. The BERT embedding provides better embedding quality than traditional embedding methods like word2vec embedding [12].

TextCNN: TextCNN is based on convolutional neural networks (CNN) to classify sentences represented by word vectors [13]. It uses multiple kernels of different sizes to perform one-dimentional convolution. This type of CNN can capture associations between adjacent words (the local correlation). In the project, we choose the kernel sizes to be $2, 3$ and 4.

Bi-LSTM: We use a one-layer bi-directional LSTM network [14] with embedding methods from BERT, followed by an attention layer and a fully connected layer. The number of hidden states is 256. Specifically, we add the attention layer after the final hidden state output for to focus on relevant parts of the input sentence, which is found to be essential for the model to provide good performance in our case.

3) Ensemble learning: We conduct the ensemble learning of the five trained classifiers with random forest to do the final classification. The final classification performance is demonstrated in Sec. IV-2.

[1] Hugging face is an AI community to provide open source NLP softwares https://huggingface.co/.

IV. Experiments

In this section, we report the results of our experiments. We describe below the setup, and then evaluate the different models in Sec. IV-1. In Sec. IV-2, based on the models, we train a Random Forest classifier to further aggregate the models and improve the performance. In Sec. IV-3, we compare the performance of BERT trained on Standard and Large Dataset. Finally, we present the result of BERT trained on a Multi-Labeled Dataset in Sec. IV-4.

Training details: We train each model for 5 epochs with batch size 64 using Adam optimizer. The initial learning rate is set as $3e-5$ for fine-tuning transformer models and $1e-3$ for downstream classification networks. A lower learning rate is important in fine-tuning the transformers, and when fine-tuning the learning rate decreases linearly from the initial value. This training technique benefits convergence of transformers and keeps the stability. To prevent overfitting, we only save the model with optimal performance on test set after each epoch.

1) Experimental results on Standard Dataset: As explained in Sec. III, we train five models on the Standard Dataset containing 1 million proper sentences and 1 million non-proper sentences to evaluate their performances.

The results of this experiment is presented in Table I.

Model	Test Accuracy
BERT	89.27%
GPT2	88.67%
BIG-BIRD	**90.26%**
BERT embedding + Bi-LSTM	86.33%
BERT embedding + TextCNN	81.40%

TABLE I: Test accuracy of five models on Standard Dataset

From the results, we can see that the transformers provide much better results than the models sequentially linking BERT embedding and either a BiLSTM or TextCNN. Specifically, BIG-BIRD provides the optimal performance, with 90.26% test accuracy. BERT and GPT2 provide similar test accuracy, 89.27% and 88.67% respectively.

2) Ensemble learning with Random Forest: In this section, we aggregate the five models (in table I) with random forest. We use a separate ablation set, instead of the train set which the models are trained on, which allows us to aggregate based on the models' decisions on unseen data. The best parameters after 10-fold cross-validation have 100 decision trees, and a maximum depth of 3. The test accuracy of the random forest reaches 90.51%, higher than the optimal accuracy among the individual models (90.26%).

3) Result on Large Dataset: In this section, we train BERT on the large dataset (5 times the size of the Standard Dataset) with less epochs (1 epoch in contrast to 5 epochs). Overall, the model is trained with the same iterations as the case with Standard Dataset. With the same training details described before (but only for one epoch), results show that training with Large Dataset provides a higher test accuracy (90.36%), compared with the accuracy trained with Standard Dataset (89.27%).

Fig. 3 presents the learning curve for BERT trained with Standard and Large Dataset (here the 'epoch' for Large Dataset acutually refers to 1/5 iterations of one epoch). We can see that, training with Standard Dataset overfits after 2 epochs and the test loss starts to increase. But when training with Large Dataset, although with fluctuation, the test loss generally keeps reducing and is lower than that of the test loss in Standard Dataset.

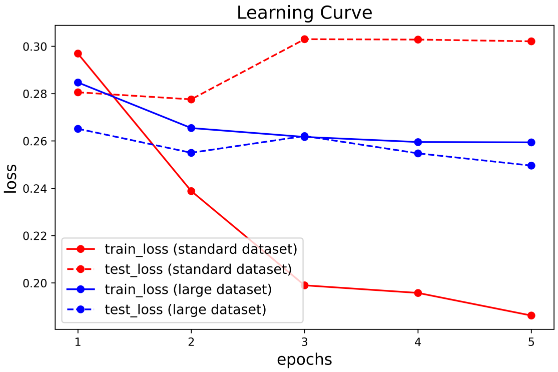

Fig. 3: Learning curve for BERT trained with Standard and Large Dataset

The results suggest that, provided with enough computational capacity, training on a larger dataset can provide better performance on test set.

4) Result on multi-label dataset: In this section, we further create a Multi-Label Dataset, which contains the same samples as the Standard Dataset, whereas the negative samples are distinctively labeled (e.g. *missing words* or *repeating words*) instead of uniformly labeled as *negative*.

We train a BERT model on this dataset, and it reached 85.01% classification test accuracy. But for this case let's look at its performance on each sample class. The precision, recall and F1-score of each class is given in Table II.

Sample Class	Precision	Recall	F1 Score	Number
Complete Sentence	0.87	0.94	0.90	109857
False Sentence Boundary	0.83	0.81	0.82	42677
False Word Recognition	0.84	0.70	0.77	21897
Missing Words	0.64	0.50	0.56	21711
Repeating Words	0.96	0.99	0.98	21781

TABLE II: Precision, Recall and F1-Score of each sample class

From the result, we can see that the simplest task is to identify repeated words in the sentences (F1-score near 0.98). Identifying complete sentences is also a relatively easy task, with a F1-score of 0.90. The hardest task for the model is detecting whether there are missing words in the sentence. It achieves only 64% precision and 50% recall on this task.

The confusion matrix is drawn in Fig. 4. From this figure, we can further see that, the classifier finds it difficult to classify between complete sentences and sentences with missing words, even though in most of the cases more than one word is missing in the erroneous sentences.

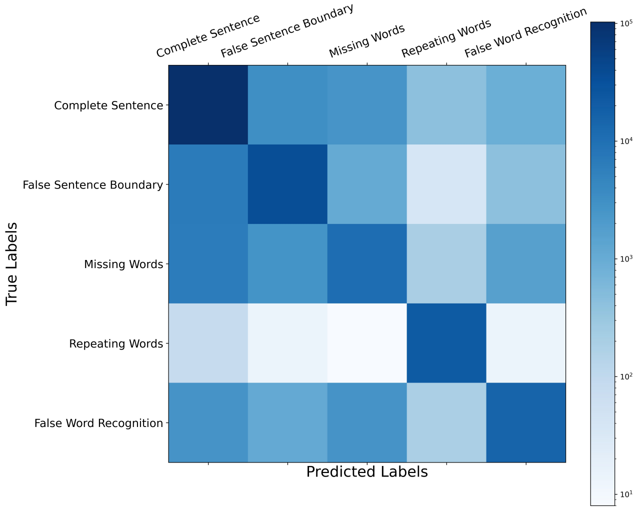

Fig. 4: Confusion matrix for BERT trained on Multi-Label Dataset

V. Discussion

From the results, we can see that transformer models are capable of providing good performance on classification of the constructed dataset for speech recognition error, reporting nearly 90% accuracy for BERT, GPT2 and BIG-BIRD.

Ensemble learning using Random Forest provides further improvements on performance, but not to a large extent. This is probably to due that the transformers (along with their embedding) share similar structures and do not diverge much on decisions. But still, Random Forest improves the optimal performance from 90.27% (BIG-BERD) to 90.6%.

The results have also shown the benefits brought by a large dataset. Training BERT on a large dataset with the same iterations provides better performance on test set.

Finally, the results of training BERT on a multi-label dataset shows that it is very hard for the model to identify whether a sentence has missing words. compared with identifying other kinds of recognition erorrs. This is understandable, because it usually requires fully understanding the logistics of a sentence to detect whether certain words are missing. For future works, special treatments might be needed to better cope with identifying missing words in sentences.

VI. Conclusion

In this project, a dataset for detecting speech recognition errors is created, where four different types of typical speech recognition errors are taken into account. Five different models are trained on the Standard Dataset, with the best test accuracy reported being 90.26% (by BIG-BIRD). A Random Forest is trained based on the five models, and further improved the test accuracy to over 90.51%. The results suggest that using state-of-art transformer models can provide good quality evaluation for detecting the errors in speech recognition systems, and provide feedback on further improvements of speech recognition systems.

References

[1] N. Agarwal, M. A. Wani, and P. Bours, “Lex-pos feature-based grammar error detection system for the english language,” *Electronics*, vol. 9, no. 10, p. 1686, 2020.

[2] Z. He, “English grammar error detection using recurrent neural networks,” *Scientific Programming*, vol. 2021, 2021.

[3] Y. Liu, E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, and M. Harper, “Enriching speech recognition with automatic detection of sentence boundaries and disfluencies,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1526–1540, 2006.

[4] Y. Liu, A. Stolcke, E. Shriberg, and M. Harper, “Using conditional random fields for sentence boundary detection in speech,” in *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL’05)*, 2005, pp. 451–458.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[6] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[7] M. Zaheer, G. Guruganesh, K. A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang *et al.*, “Big bird: Transformers for longer sequences.” in *NeurIPS*, 2020.

[8] A. Thompson, “All the news: 143,000 articles from 15 american publications,” =https://www.kaggle.com/snapcrack/all-the-news, 8 2017.

[9] N. Reimers and I. Gurevych, “Making monolingual sentence embeddings multilingual using knowledge distillation,” *arXiv preprint arXiv:2004.09813*, 2020.

[10] W. Foundation. Wikimedia downloads. [Online]. Available: https://dumps.wikimedia.org

[11] K. Gopalakrishnan, B. Hedayatnia, Q. Chen, A. Gottardi, S. Kwatra, A. Venkatesh, R. Gabriel, D. Hakkani-Tür, and A. A. AI, “Topical-chat: Towards knowledge-grounded open-domain conversations.” in *INTERSPEECH*, 2019, pp. 1891–1895.

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[13] Y. Zhang and B. Wallace, “A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification,” *arXiv preprint arXiv:1510.03820*, 2015.

[14] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with lstm,” *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.