Report for visualizing BERT embeddings

Zheng Tang

December 17, 2019

1 Introduction

Deep contextualized word representations like BERT or ELMo have achieved great success in many NLP fields. But not like the traditional embeddings like word2vec or glove, BERT vectors are based on its surroundings words, which means the embedding of the same word may differ from sentence to sentence. The good news of this kind of embeddings is they include the contextual information in the vectors and the experiments showed that it really improve the results for many tasks. But the bad news is it needs more resources and time to train the model and the embeddings are not re-usable since the embeddings are different in different sentences. In this project, I built an interactive UI for visualizing the BERT embeddings.

2 Related Works

There are plenty of research on BERT since it released with state-of-art performance on so many tasks. [Devlin et al., 2018] describe a method to check if the contextual embeddings capture the syntactic information in the sentence. They trained a transformation matrix between two embeddings and the squared L2 distance will correspond to their distance in parsing tree. [Coenen et al., 2019] find that the pre-trained BERT encode some syntatic and semantic information by probing BERT's geometry representations

3 Data

We train our model using SNLI corpus, SNLI is a collection of English sentence pairs manually labeled for balanced classification with the labels entailment, contradiction, and neutral, supporting the task of natural language inference (NLI). We sampled 10,000 pairs from the whole corpus.

4 Method

We use BERT model provied by pytorch transformers. We first feed all sentence pairs to BERT and get the pre-trained embeddings for all words in our dataset.

For SNLI task, we put a feedforward layer on top of the BERT model. We use CrossEntropyLoss as the loss function and feed all sentence pairs to train this model.

After we get the model, we feed the sentence pairs to our model again and extract the fine-tuned BERT embeddings from our model. So we get two set of word vectors.

To visualize the vectors, we use UMAP to map the vectors to 2-d.

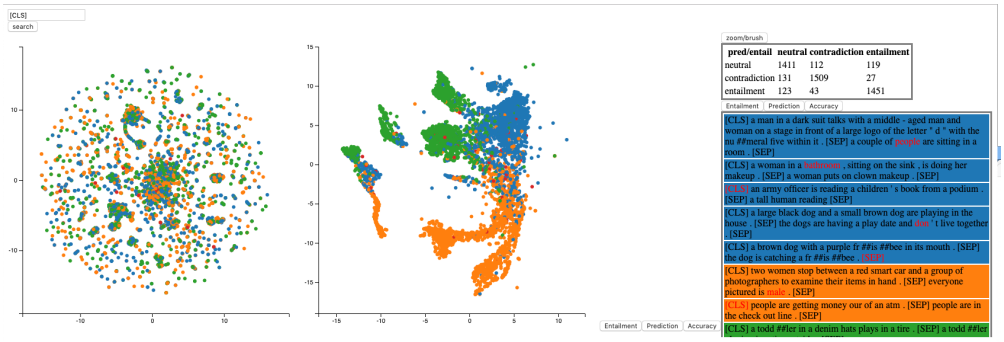


Figure 1: The overview of our UI, we sampled the points and show them in the scatterplots, the plot in the left is the word vectors before fine-tuning and the right one is after fine-tuning. The color indicates the label of the sentence pairs this word belongs to. Green means entailment, orange means contradiction and blue means neutral. We put a search bar in the UI, people can type any word they interested in and filter the scatterplots to only show the vectors for searched word. The statistic table shows statistics for selected points, if no selection, it will show the statistic for all words. We also sampled 10 sentences for selected word.

5 UI design

As we describe in figure 1, Our UI has four main parts: the search bar, scatterplots for word vectors, statistics for SNLI task and sentences for selected word.

Also we put interactive functions to the scatterplots, People can click or brush a rectangle to select points in them, poeple can also zoom in/out the plots. For plot in the right-hand side, people can change the color from gold label to predicted label, they can also change the color based on the prediction correctness(green for positive and red for negative) as shown in figure 2

For the statistic table, we can click each cell and filter the scatterpolt based on the selection. As shown in figure 3, we clicked the number of entailments predicted as contradictions.

We can click the sentence and show all words vectors in the scatterplots. We use same color and switch button for sentences as for right plot

5.1 Observations

Mainly we observe three interesting facts in this project:

(1) We can see that before fine-tuning, the vectors locates in a random place, and for each word, the vectors cluster by its semantic meanings, for example, vectors for dog for hot dog locate in a different cluster from dog for animals.

(2) After fine-tuning, the vectors are clustered by the task label and lose its semantic information which we believe it is kind of overfeeding because even the task-free tokens like . or [SEP] are also clustered based on the task.

(3) Although we don't feed any information to our model about the meaning of entailment, contradiction and neutral, fine-tuned BERT some-

how learn that neutral is a status between entailment and contradiction.

6 Different Settings

We also visualize the [CLS] tokens alone using UMAP as shown in fig 4. So that we can get more sentence level information. And we notice that the observations we found for all tokens still hold for [CLS] tokens. And since [CLS] tokens are more related to the task and the sentences, we can use these plots for debugging purpose.

6.1 Debugging

(1) We first check the outlier nodes as shown in fig 5. We find that BERT separate the incomplete sentences and sentences with negotiation from other sentences.

(2) We then check the sentences which we predicted incorrectly. We find that the error in boundary area is easier for us to correct compared to the error in other area. Like the example we showed in fig 6. We can easily tell that the left example is an entailment, but since our model doesn't learn that saxophone is an instrument, it predict this as a contradiction. But in right-hand side sentences, we get confused as human by the term "his picture", does it mean the picture of himself, or the picture that he takes?

According to this findings, we think we can use this tool for debugging purposes. For some errors in boundary area, we think we can solve them by introducing BERT-large and some knowledge base.

References

[Coenen et al., 2019] Coenen, A., Reif, E., Yuan, A., Kim, B., Pearce, A., Viégas, F., and Wattenberg, M. (2019). Visualizing and measuring the geometry of bert. *arXiv preprint arXiv:1906.02715*.

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

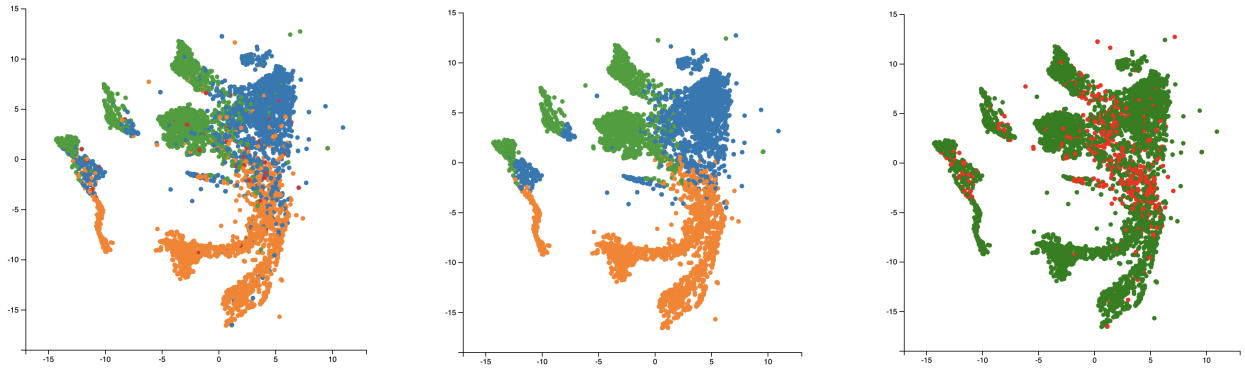
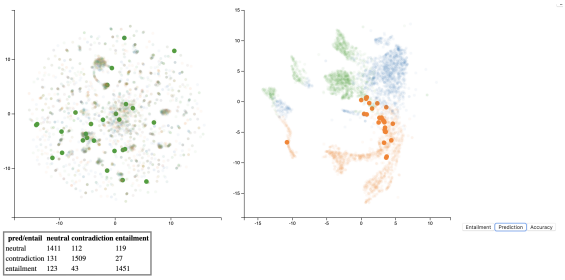


Figure 2: scatterplot color changing



pred/entail	neutral	contradiction	entailment
neutral	1411	112	119
contradiction	131	1509	27
entailment	123	43	1451

Figure 3: Shown only words in an entailment pair but labeled as contradiction

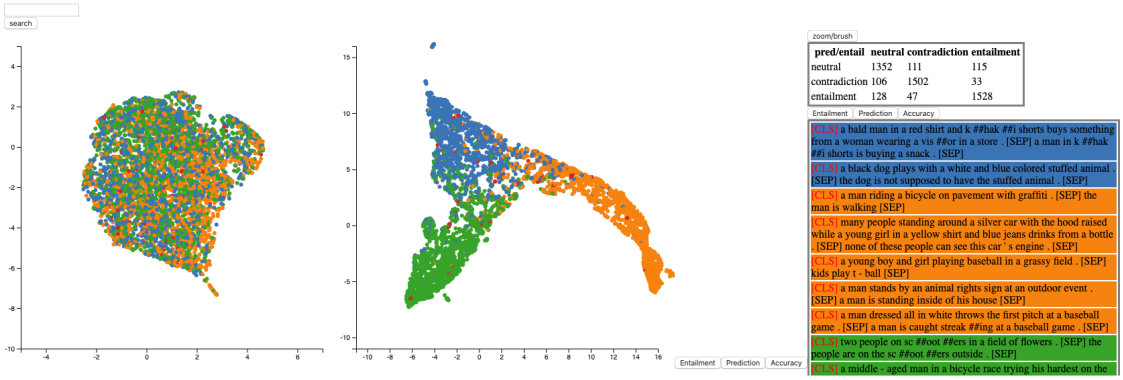


Figure 4: Points for only [CLS] tokens

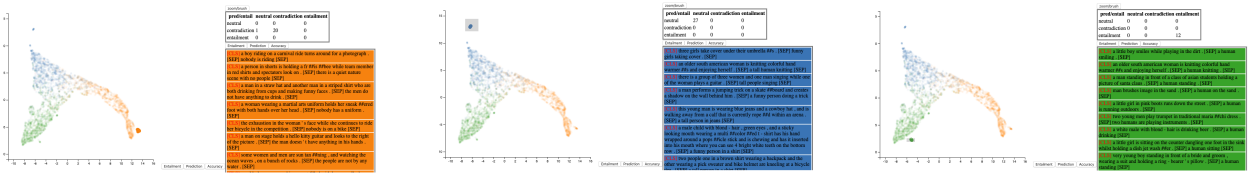


Figure 5: Outlier Analysis

Figure 6: Error Analysis