NLC2CMD Report From jb Team

Denis Litvinov DENIS.LITVINOV@JETBRAINS.COM
Gleb Morgachev GLEB.MORGACHEV@JETBRAINS.COM
Artem Popov ARTEM.POPOV@JETBRAINS.COM
Nikolai Korolev NIKOLAI.KOROLEV@JETBRAINS.COM
Dmitrii Orekhov DMITRII.OREKHOV@JETBRAINS.COM

Keywords: Seq2Seq, Text Augmentation, Transformers

1. Introduction

NLC2CMD competition presents a task to predict bash command from natural language query.

2. Data Description

Participants are given with a training dataset of 10k samples. After closer examination, some examples were found to be duplicates. We found that the dataset contains not only pipes but also nested commands. Though, deeply nested commands were rare.

3. Data Augmentation

Because the training dataset is so small, we decided to heavy augment original data through backtranslation (Alzantot et al. (2018)). We used fairseq pre-trained models and experimented with sampling with temperature. Also, we created filters, so augmented texts should be in a certain BLEU score range. It should be noted that augmentation was applied only to text invocations. So multiple entries with the same bash commands were generated, thus introducing some leak into data.

4. Data Generation

Augmentation wasn't enough, so we generated new examples from manpage data. Bash command was created by concatenating utility name and several randomly sampled flags. Text query was created by concatenating utility synopsis and truncated flag descriptions. Also, some adjustments were made with common pipelines endings like "wc" or "grep", where it is easy to modify text description ("how many .. ", "get all ..." correspondingly)

5. Data Preprocessing

Because we used transformers both for classification and sequence prediction, the length of the input sequence was crucial. Text preprocessing included removing stopwords from nltk and stemming by SnowballStemmer. To make the seq2seq task easier, bash commands also were preprocessed. All arguments were replaced with "ARG" placeholder. All arguments which presence do not change the accuracy metric were removed.

6. Solution

After examining several approaches, we finally decided to use the ensemble of 2 models, trained separately on the same dataset.

The first model is a classifier that predicts the first utility in a sequence. The first model is used to select an appropriate context, concatenate it with text query and push to the second model.

The second model is a seq2seq transformer (Vaswani et al. (2017)), which is trained to predict the entire bash command from text query and the context. Context is created by concatenating utility synopsis and truncated flag descriptions. The context model was trained in a teacher forcing mode. Predictions from the context model are obtained by classic beam search with width 5.

The final prediction is made by blending predictions from both models: for each top-5 utility predictions the context model and beam search are applied. Resulted 25 candidates are sorted by their joined probability and top-5 most probable are selected for an answer. We found that when estimating joined probability blending probabilities of 1st and 2nd model gives better results than just adding them up.

As expected, the heavily augmented dataset introduced some leaks, so relying only on local validation score is not trustworthy, because the model overfits. But model weights on some earlier epochs give better results.

7. Conclusion

Here we presented our solution to the NLC2CMD challenge, which we believe have practical applications and can make the life of some developers easier. To mitigated different combinations of pipes and nested commands, we selected a rather general approach in model architecture. We think that introducing more data model performance can be improved.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani B. Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. *CoRR*, abs/1804.07998, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.