LaTeX Handle This House Property Dialog Dataset Better

Haozhe Yang
202015102
sailist@outlook.com

Abstract

Dialog matching task is to estimate the matching degree of an answer to a question. To extract features from texts, an encoder-decoder structure is aften used. In this paper, I introduced the methods I used to learn this task on a dialog dataset in house property area from the 2020 CCF contest. First, I designed different models to find a way to extract features from texts better. Then, I also tried some methods to alleviate the problem about long-tail distribution and class-imblance. Finally, my methods reached 0.7679 after fine-tuned with 5-fold cross validation.

1. Introduction

Dialog systems are becoming more and more important area in natural language processing. An important research topic in dialogue systems is dialog matching. As illustrated in Figure[], given a pair of question and answer sentence, dialog matching aims to judge whether the anwser is precisely enough to this question. This is a hard task as there may be a huge semantic gap between questions and their answers.

Recently, different works in different area has been proposed to handle this problem. [4] proposed a method by using latent space and map different types of answer in this latent space around context. [9] define a new conception, mutual persona perception, to model the dialog context. Besides, some methods found that better results can be reached by using a pre-trained from unsupervised large text datasets and well-designed model to extract feature from text first. Like Transformer [5],GPT-1 [3]. Meanwhile, text augmentation is another direction, which helps people train robust model.

In this paper, to get a better result on the dialog dataset in house property area from the 2020 CCF contest, I try different methods, including data-preprocessing, model designing, loss constructing, etc.. My key contributions of this work are:

- Propose a pipeline framework for handling dialog matching problem.
- I try different methods to reach a better results in each part.
- My experiment reaches 0.768 on test dataset of this contest.

2. Related work

2.1. Data Augmentation

Automatic data augmentation is commonly used in computer vision and speech. It's useful to help people train more robust models, especially when using smaller datasets. Some previouse works have proposed some techniques for data augmentation in NLP. [2] generated new data by translating sentences into another language and back into the original one. Other work augmented data by adding noise [11] and replace word by synonym [7]. Due to these work is hard to use in practice because of the high cost of implementation problem, [6] presented a simple set of univsal data augmentatoin techniques, which can efficiently imporve the result, particularly for smaller datasets.

2.2. Pre-train model

Unsupervised pre-training is a common method to find a good initialization instead of train a randomly initialized model. Previous works explored this technique in image classification, but recent years, some research demonstrated that pre-training also works in NLP. Like BERT [5], GPT-1[3], GPT-2[1]. But use this model is also a trade-off between performence and memory/time cost.

2.3. Class-imblance

Most read-world datasets do not have exactly equal number of instances in each class. A small difference often does not matter, but when the data has a long-tail distribution, it aften hurt the model performence. This phenomenon is commonly seen in different area, such as Medical image processing, object segmentation, etc.. In dialog dataset, responses like 'yes' is the most common things, so there is also a class-imblance problem. To handle this problem, different methods have been proposed. [10] constructed a loss named Focal-loss to down-weights the loss assigned to well-classified examples. [8] proposed a meta-learning method, to generate weights for current batch samples by using a small clean class-blanced validation dataset. All these methods reached good results on their areas.

3. Method

Before presenting the model, I first provide the problem formulation. Suppose that we have a dialog dataset $\{Q, A, C, Y\}$ for matching, I denotes $Q = \{q_1, q_2, ..., q_m\}$ as a question set, $A = \{a_1, a_2, ..., a_n\}$ as a answer set, $C = c_1, c_2, ..., c_n, c_i \in [1, m]$ to indicate which q_j is a_i answer for, which means (q_{c_i}, a_i) can be a pair or a sample. $Y = y_1, y_2, ..., y_n, y_i \in \{0, 1\}$ as a label set, to indicate whether a_i can be a good answer to q_{ci} . Then the dataset can be reformulated to $D = \{(q_{ci}, a_i, y_i)_i\}_i^N = 1$.

Given a dataset D , the objective function of training neural networks can be represented as:

$$\underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} L\left(y_i, M\left(F\left(q_i; \theta_q\right), F\left(a_i; \theta_a\right); \theta\right)\right)$$

where F denotes the encoder, and M with parameter θ is defined to project features from the concat of question and answer to 2-dimention vector, which can be used for classification.

3.1. Encoder

Denote the map function as Transformer is $T(\cdot)$, for each input x the output $s = T(x)$, where $s \in R^{T,F}$, T is timestamp,and F is the dimention of the feature in each timestamp. Since $T \times F$ is a high dimention vector, It's better to project it into a lower dimention space by another map function. In this paper, I tried three methods.

Simple MultiLayer Perceptron Simpliest way to reduce the dimention is to add multilayer perceptron. First, I build 4 vectors with F -dimention, the first two vectors are the first and the last features alone the timestamp axis of s , the last two are generated from average pooling and max pooling operation separately. Then I concatate these four vectors and continue reducing its dimention by a dense layer.

Resnet blocks I found that the first method does not work well, and one possible reason might be the loss of a lot of information when using pooling operation. To extract more useful information from feature s , I build a 1D resnet by replacing all 2D operations in original Resnet[] into 1D oueartions, which can be used to extract feature from text features.

An Ensemble A better feature can be extracted by an ensemble of different extractors. In this version, I build a two-branch decoder, where one of these is still a 1D resnet structure, and another is a multilayer perceptron layer alone the timestamp axis. Feature from two branchs are also concatenated and then a dense layer is used.

Finally, I concatenate the feature from question q_i' and answer a_i , and map it to 2-dimention vector by adding a linear project layer. Then a cross-entropy loss function can be constructed. I further explored Focal-loss[] to alleviate the affect caused by class-imblance.

4. Experiments and Results

4.1. Dataset

In this paper, I evaluate our methods on a dialog dataset in house property area from the 2020 CCF contest. It consists nearly 6K questions and 20K answers with labels for train, and 14K questions and 53K answers without labels for test. Due to its class-imblance and less train data that test, it's a challenging task.

When preprocess the data, the sentence need to be converted into token id, and be padded or truncated to the same length. The length, which is a hyperparamter in our experiments, is also a trade-off. Larger length may improve the performance, and smaller can save more time. Finally, I set this value to 25.

4.2. Implementation Details and Results

First, I train my model using SGD with momentum of 0.9, a weight decay of [todo], and a batch size of [todo]. The network is trained for 20 epochs. Test results are given by a 5-fold cross validation, which is a procedure used to estimate the skill of the model on new data. I find that most hyperparameters do not need to be heavily tuned. I first froze the parameter in BERT and treat it as a pure feature extractor to save memory, which reached a worse result, 0.69. Then I choice to train the whole model on a full TITAN XP, and I finally reached 0.7679.

5. Conclusion

In this paper, I mainly explored three methods for extracting useful method and reducing dimention from high-dimention vectors at the same time. I then mixed focal-loss and meta-learning method to try to handle the class-imblance problem. The results show that well-designed encoder can utilize the feature better that baseline, which only use one linear project layer. In the future, it would be interesting to further investigate better method to extract more useful information.

6. Reference

References

[1] R. C. A. Radford, Jeffrey Wu. Language models are unsupervised multitask learners. *arXiv:1902.11205*, 2019. 1

[2] M.-T. L. Adams Wei Yu, David Dohan. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv:1804.09541*, 2018. 1

[3] T. S. Alec Radford, Karthik Narasimhan. Improving language understanding by generative pre-training. *Openai*, 2018. 1

[4] X. Gao, S. Lee, and Zhang. Jointly optimizing diversity and relevance in neural response generation. *arXiv:1902.11205*, 2019. 1

[5] K. L. Jacob Devlin, Ming-Wei Chang. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*, 2018. 1

[6] K. Z. Jason Wei. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv:1901.11196*, 2019. 1

[7] S. Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv:1703.02573*, 2017. 1

[8] B. Y. Mengye Ren, Wenyuan Zeng. Learning to reweight examples for robust deep learning. *arXiv:1803.09050*, 2018. 2

[9] B. C. Qian Liu, Yihong Chen. You impress me: Dialogue generation via mutual persona perception. *arXiv:2004.05388*, 2020. 1

[10] R. G. Tsung-Yi Lin, Priya Goyal. Focal loss for dense object detection. *ICCV*, 2017. 2

[11] J. L. Ziang Xie, Sida I. Wang. Data noising as smoothing in neural network language models. *ICLR*, 2017. 1