1 Abstract

Transitioning between topics is a natural component of a dialogue system. The goal of our task is to guide users to the target topics or services through interacting with our bot in a dialogue. We first collect our data from `blended_skill_talk` and dialogues generated from `blenderbot`. We then explore different strategies to generate the training data for our generation model. Our work shows good performance on naturalness, relevance, and aggressiveness and has 0.819 keyword hit rate on the test set.

2 Introduction

2.1 Data

The test datasets are `blended_skill_talk` from hugginface. There are 980 sentences in the test set, and our task is to train a bot to generate fluent and human-like responses to perform a topic transition from this sentence.

This dataset is from a paper "Can You Put it All Together: Evaluating Conversational Agents' Ability to Blend Skills", which states that this dataset is aimed to train a agent with different blend skills.

2.2 Model

We use T5, text-to-text transfer transformer, which is a sequence to sequence model with encoder-decoder frame and pre-trained on dataset C4, Colossal Clean Crawled Corpus. The pre-trained model is loaded from hugginface.

3 Approach

3.1 Data

At first, we collected some data from `blended_skill_talk` and some other dialogues generated from two simulators, and we filtered the result to ensure that there are some sentences containing the keywords in the dialogues. After that, we came up with four different types of training data:

1. The inputs are all the sentences before the sentences containing some keywords, which is the output (we call this **target sentences**). In this case, we hope that our bot can learn to speak keywords. (2659 data)
2. The inputs are all the sentences before the sentences, and the output is the sentence before the target sentences. In this case, we hope that our bot can lead the simulator to say the target sentences. (2659 data)
3. This is a variation of type-I training data. The output can be all the sentences before the target sentences, and the input is the sentences from the dialogue start to that output. We think that our bot should learn how to lead the conversation from the start to the target sentence. In this case, the output sentence might not contain some keywords. (7417 data)
4. This is also a variation of type-I training data. The input is the one, two, three, ... sentences just before the target sentences. We think that every combination of sentences before the target sentences should lead our bot to say the target sentences. (8082 data)

We generated these four types of training data and trained four models for each case, and we obtain the results that type-I had 0.82 hit rate, type-II can reach 0.9 hit rate, while type-III data can reach 0.8 hit rate. Even though type-II data can reach the highest hit rate, but the sentences it generated are weird. Therefore, we decided to train our model with type-III data.

At this point, we had a basic model, so we decided to give it more data and enlarge the size of the pretrained model. We found another dataset `daily_dialog` from huggingface. This dataset is some daily conversation between

two people. We again filtered the sentences to get some dialogues with keywords and did the type-III transform on it to generate the training dataset we want. We combined the training datasets and the final training datasets containing 16033 data.

3.2 Model

3.2.1 tokenizer

Different from Bert tokenizor with special token [SEP] to separate sentences, T5 tokenizor don't have a specific token to do that. We just simply use space to connect input sentences.
For padding and truncate, we use small *max_len* and *max_target_len* to truncate input and target sentences in order to speed up training process; however, the outcome become incomplete. We finally use larger ones to avoid that. For more detail, please check 4.1.

3.2.2 T5-small & T5-base

First, we train T5-small on four different datasets to roughly evaluate the training time and pros and cons of different datasets. Then we decide to enhance both our model and dataset to obtain more robust outcome by train T5-base on the bigger dataset mentioned above.
Following is our model configs :

```
{
  "_name_or_path": "t5-small",
  "architectures": [
    "T5ForConditionalGeneration"
  ],
  "d_ff": 2048,
  "d_kv": 64,
  "d_model": 512,
  "decoder_start_token_id": 0,
  "dropout_rate": 0.1,
  "eos_token_id": 1,
  "feed_forward_proj": "relu",
  "initializer_factor": 1.0,
  "is_encoder_decoder": true,
  "layer_norm_epsilon": 1e-06,
  "model_type": "t5",
  "n_positions": 512,
  "num_decoder_layers": 6,
  "num_heads": 8,
  "num_layers": 6,
  "output_past": true,
  "pad_token_id": 0,
  "relative_attention_num_buckets": 32,
  "task_specific_params": {
    "summarization": {
      "early_stopping": true,
      "length_penalty": 2.0,
      "max_length": 200,
      "min_length": 30,
      "no_repeat_ngram_size": 3,
      "num_beams": 4,
```

```
      "prefix": "summarize: "
    },
    "translation_en_to_de": {
      "early_stopping": true,
      "max_length": 300,
      "num_beams": 4,
      "prefix": "translate English to German: "
    },
    "translation_en_to_fr": {
      "early_stopping": true,
      "max_length": 300,
      "num_beams": 4,
      "prefix": "translate English to French: "
    },
    "translation_en_to_ro": {
      "early_stopping": true,
      "max_length": 300,
      "num_beams": 4,
      "prefix": "translate English to Romanian: "
    }
  },
  "torch_dtype": "float32",
  "transformers_version": "4.17.0",
  "use_cache": true,
  "vocab_size": 32128
}
```

```
{
  "_name_or_path": "./t5-base",
  "architectures": [
    "T5ForConditionalGeneration"
  ],
  "d_ff": 3072,
  "d_kv": 64,
  "d_model": 768,
  "decoder_start_token_id": 0,
  "dropout_rate": 0.1,
  "eos_token_id": 1,
  "feed_forward_proj": "relu",
  "initializer_factor": 1.0,
  "is_encoder_decoder": true,
  "layer_norm_epsilon": 1e-06,
  "model_type": "t5",
  "n_positions": 512,
  "num_decoder_layers": 12,
  "num_heads": 12,
  "num_layers": 12,
  "output_past": true,
  "pad_token_id": 0,
  "relative_attention_num_buckets": 32,
  "task_specific_params": {
    "summarization": {
```

```
      "early_stopping": true,
      "length_penalty": 2.0,
      "max_length": 200,
      "min_length": 30,
      "no_repeat_ngram_size": 3,
      "num_beams": 4,
      "prefix": "summarize: "
    },
    "translation_en_to_de": {
      "early_stopping": true,
      "max_length": 300,
      "num_beams": 4,
      "prefix": "translate English to German: "
    },
    "translation_en_to_fr": {
      "early_stopping": true,
      "max_length": 300,
      "num_beams": 4,
      "prefix": "translate English to French: "
    },
    "translation_en_to_ro": {
      "early_stopping": true,
      "max_length": 300,
      "num_beams": 4,
      "prefix": "translate English to Romanian: "
    }
  },
  "torch_dtype": "float32",
  "transformers_version": "4.17.0",
  "use_cache": true,
  "vocab_size": 32128
}
```


4 Experiment/Evaluation

4.1 max truncate and max target length

We first tested different length input(max_len) and generated(max_target_len) sequence have. We realized that once we lower the max_len and max_target_len, the outcome sentences were either too short or couldn't complete what it was about to say. Thus, we decided to let $max\_len = 600, max\_target\_len = 150$ since we could have eight sentences at most and wanted bot could complete its sentences.

4.2 model

We compared the two different models t5-small and t5-base. We discovered that while t5-base was more time-consuming on training, it could generate more natural sentences. Dialogs generated from t5-small are tend to have with multiple and redundant questions in one sentence, which is annoying and far from natural conversation.

4.3 Generate Strategies

After determined which model is better, we then tested different generate stragies ranging from beam search, top-k and top-p sampling, and temperature.

4.3.1 Beam Search

We first tried the different number of beams(*num_beams*) on beam search. As we had predicted, the higher *num_beams* was, the more general words it would contain. After several comparison, We chose 30 as our submitted outcome.

4.3.2 Top-k Sampling

We compared the outcome generated from different number of sampling(5∼100). As it turned out, if the top-k was too high or too low, the most of the sentences from our bot would contradict what it have said, which is unnatural.

4.3.3 Top-p Sampling and Temperature

We tested different combinations of top-p(0.8, 0.85, 0.9) and temperature(0.8, 1.0, 1.2). We concluded that high temperature would cause most of the sentences contained more general words like thanks, welcome, and like while high top-p would make sentences unnatural and disjointed from the dialog and original topic. Thus, we chose dialogs generated from (top-p, temperature) = (0,85, 0.8), (0.8, 0.8), (0.8, 1.0) as the final three outcome to compare with other strategies.

5 Conclusion

After experiments and analysis, we selected t5-base as our main model with $num\_beams = 30$, $max\_len = 600$ and $max\_target\_len = 150$, and trained it with the third data generation (output be all the sentences before the target sentences, and the input is the sentences from the dialogue start to that output), with these training approach, our model has 0.819 keyword hit rate on the test set, and also ranks 8, 3, 6 and 6 in Naturalness, Relevance, Aggressiveness, Overall of human evaluation.

6 Work Distribution

1. Data collection and generation: 洪國瀚、方泓傑、王秀軒
2. Model design and training: 黃宏鈺、葉乃瑄
3. report and presentation: 黃宏鈺、葉乃瑄、方泓傑、王秀軒、洪國瀚