Text generation experiments for Dialdoc21 Shared Task: Comparing SacreBLEU scores of generated text from pre-trained transformers using dialogue history and document context

Subin Park
Centre for Doctoral Training in Interactive AI, University of Bristol
`subin.park@bristol.ac.uk`

Abstract

This report is about my response generation experiments for Dialdoc21 Shared Task. I generate text using dialogue history and document context from the doc2dial dataset. Firstly, I use a GPT2 text generation pipeline. Secondly, I conduct the same task with another transformer model, XLNet, to compare the results. I evaluate the generated text using SacreBLEU scores. Finally, I summarise my work and suggest fine-tuning as a future work that could further improve the performance of pre-trained transformer models.

1 Introduction

The number of recent works that model document-grounded dialogues is increasing as a major type of information-seeking dialogue systems. The research progress in real-life applications of dialogue systems have underlined the importance of a better understanding of the inter-relations between conversations and documents (Feng et al., 2020; Ishii et al., 2021). Feng et al. (2020) developed the doc2dial dataset and explain how this can be used for implementing and evaluating a dialogue system that can help users seeking information. Alongside great performing text generation models have been developed which can produce a human-like quality of texts with short prompts.

I generate an agent utterance using the retrieved knowledge with transformer models. My research question asks what influences the performance of pre-trained text generation models. To answer this question, I compare the results from two different transformer models with and without feeding the entire dialogue history and/or indicating the user utterance turn in the dialogue.

I use pre-trained transformer pipelines for the text generation task. Pipelines can be run with as little as two lines of code, hence easier to use than direct models. Transformer pipelines are so versatile that they can be used for various tasks such as question answering, language modelling, summarisation and translation as well as classification tasks.

The text I generate in the first stage does not use any document context. The GPT2 pipeline (Radford et al., 2018) generates answers purely from the single input question, a user utterance. Next, when feeding the context document to my second transformer method, XLNet (Yang et al., 2019), the pre-trained model sometimes generates better-performing answers in terms of SacreBLEU scores and human evaluation. After running experiments, I find that XLNet using the document context performs better than GPT2 without the context. However, feeding dialogue history to these models does not improve the SacreBLEU scores. Furthermore, indicating the user utterance at the prompt as input does not always help to generate more useful answers.

This leads us to the next challenge, to fine-tune these pre-trained transformer models with the doc2dial dataset for the Shared Task to improve the performance even more. I describe the process of my attempt at fine-tuning transformer models in the future work section of this report. Finally, my code running pre-trained transformers with doc2dial dataset is available in my GitHub here: `https://github.com/travelingsubin/doc2dial-textgeneration/blob/main/DN_coursework.ipynb`.

2 Background

The Dialdoc21 Shared Task is composed of two sub-tasks: Knowledge Identification; to retrieve the knowledge from the documents and Response Generation; to generate an agent utterance using the retrieved knowledge. For more about the task, see `https://doc2dial.github.io/workshop2021/shared.html`. The doc2dial dataset consists of 4793 dialogues and 487 documents in total. Each document is annotated with domain, title, document content and span information. Each dialogue consists of an information-seeking conversation between a user and an agent, where the agent's responses are grounded in FAQ-like web pages. The dialogues contain document id and domain. Each dialogue turn contains dialogue scenes, which includes role, dialogue act and grounding reference. The dataset is downloadable at `https://doc2dial.github.io/data.html`.

As I focus on Subtask 2, I assume the output from Subtask 1, a text span, is given as a form of 'answers' in the 'doc2dial_rc' sub-dataset. Subtask 1 is formulated as a span selection problem that requires identifying the knowledge in an associated document. Hence, the goal of Subtask 2 is to generate a response using the knowledge extracted from the document. The input is the dialogue history, current user turn and the extracted text span. The dialogue history can be understood as all turns of the user and agent utterances for each conversation until the current turn.

One example of input is the last user utterance in a dialogue, asking a question to an agent such as "Hello, I forgot to update my address, can you help me with that?". The output can be a few sentences like "You must report a change of address to DMV within ten days of moving. That is the case for the address associated with your license, as well as all the addresses associated with each registered vehicle, which may differ". The context from the associated document can often help generate more useful answers.

Two transformer models that I used for my text generation experiments are:

1. **Transformer GPT2 text generation pipeline.** GPT2, developed by OpenAI, is a direct scale-up of GPT, with more than 10 times the parameters and trained on more than 10 times the amount of data. Original GPT is the first autoregressive model based on the transformer architecture, pre-trained on the Book Corpus dataset. More detailed information about GPT2 is available at `https://huggingface.co/docs/transformers/model_doc/gpt2`.

2. **Transformer XLNet.** XLNet is an extension of the Transformer-XL model and is released by Google/CMU. This autoregressive model learns bidirectional context by maximising the expected likelihood over all permutations of the input sequence factorisation order. Information about the model is published at `https://huggingface.co/docs/transformers/master/en/model_doc/xlnet`. Codes for application is available at `https://github.com/huggingface/transformers/blob/master/examples/pytorch/text-generation/run_generation.py`.

Both transformers are autoregressive models. Autoregressive models usually rely on the decoder part of the original transformer and use an attention mask so that at each position, the model only needs to look at the tokens before the attention heads (More information about general features of transformer models is at `https://huggingface.co/docs/transformers/model_summary`).

3 Experiment overview

After loading the datasets, I generate text in different conditions using transformers where inputs include the dialogue history or the last user turn and the outputs are the ground truth text spans. I focus on transformers that perform the text generation task. I choose to use transformers because they support numerous models and I can easily customise different models with low computation costs.

I run text generation experiments using two dialogue examples, Top DMV Mistakes and Dial-in Search Accounts, from the dataset. Then I evaluate the results upon the SacreBLEU score. Some of the advantages of SacreBLEU include that it automatically downloads common Workshops on Statistical Machine Translation (WMT) test sets and processes them to plain text and also produces a short version string that facilitates cross-paper comparisons. This metric is as easy as only comparing the generated response and the reference text. In this shared task, the reference text is from 'answers' in doc2dial_rc' sub-dataset which can be considered as a golden standard. For more information about how the metric works, see README.md at (`https://github.com/mjpost/sacreBLEU`).

4 Results

I generated text with GPT2 and XLNet transformers with and without dialogue history and the user indicator. When feeding the dialogue history, the user indicator is included in the input of the transformer model. Therefore, I skip the experiment without the user indicator in this condition. The SacreBLEU scores for each experiment vary as follows.

SacreBLEU score (min/max)				
Transformer model	Example	w/o dialogue history	w/o dialogue history + user indicator	with dialogue history
GPT2	Top 5 DMV Mistakes	0.95/2.70	0.70/1.52	0.99/1.24
GPT2	Dial-in Search Accounts	2.02/2.91	2.63/3.67	0.87/2.19
XLNet	Top 5 DMV Mistakes	2.60	1.20/1.35	2.19
XLNet	Dial-in Search Accounts	3.38	2.41	2.59

Table 1. SacreBLEU score comparison between GPT2 and XLNet with two dialogue and document examples depending on different input settings

Table 1 shows that GPT2 performs better without dialogue history than with the history. Yet, XLNet performs better than GPT2 in general, either with our without dialogue history. That is because XLNet knows the context of the associated document of all dialogues. However, interestingly, the highest score is observed with GPT2 without dialogue history plus indicating user utterance at the input.

The reason why GPT2 can sometimes perform as good as XLNet even without the associated document context for the dialogue is that GPT2 is trained with 8 million web pages, a very large corpus of about 40 GB of text data. This result arouses my curiosity and makes me wonder whether fine-tuning to a specific dataset can outweigh the strong performance of a transformer model that is trained with a vast amount of data.

5 Conclusion and future work

Although it is an interesting finding that the SacreBLEU score varies with different conditions of text generation transformer inputs, the scores from my experiments are lower than the performance of the strongest teams on the leaderboard. The leaderboard is available at `https://eval.ai/web/challenges/challenge-page/793/leaderboard/2173`. Fine-tuned transformer models can perform better because this allows parameters to adjust to the specific dataset and the context of the task.

Feng et al. (2020) published their fine-tuning codes at `https://github.com/doc2dial/sharedtask-dialdoc2021`. With this, using Google Colab, I tried to fine-tune a pre-trained transformer model, BART (large-sized model, fine-tuned on CNN Daily Mail, more information about the model at `https://huggingface.co/facebook/bart-large-cnn`). However, their original code runs 10 epochs with batch size 1 and max source length 1024, which is computationally costly. Thus, I reduced the number of epochs and max source length, and increased the batch size, so that the Colab does not crash in the middle of the training. Eventually, I was only able to run a single epoch. For example, running 10 epochs with the given condition takes about 62 hours. Moreover, changing batch size from 1 to any higher number crashes as Google Colab free version provides only 11.83 GB of GPU to an individual user, while 10.57 GB is reserved by PyTorch to run transformers. In the end, only 1.81MB for training was left for larger batch sizes. Therefore, I tried to run 5 epochs with batch size 5 and max source length 300 but this training stopped after 1 epoch due to OverflowError which is also raised by one of the Subtask challengers `https://github.com/doc2dial/sharedtask-dialdoc2021/issues/2`. In fact, to fix this error, one would need to convert the `-100` tokens in label ids to `tokenizer.pad_token_id`. In the end, I learned the entire process of fine-tuning a transformer model although I was not able to run a full experiment due to limited GPU capacity for this report. My experiments on fine-tuning is available in my GitHub at `https://github.com/travelingsubin/doc2dial-textgeneration/blob/main/finetuning_practice.ipynb`

Fine-tuning transformer models is done by other teams who participated in this shared task. For example, CAiRE (Ishii et al., 2021) authors pre-trained a BART model with a WoW dataset as this additional dataset contains richer information. Then the authors fine-tuned this pre-trained model with the doc2dial dataset. They achieved the SacreBLEU score up to around 38, which is a great performance as ranked at 3rd of the task competition as of January 2022. Their codes are available at `https://github.com/HLTCHKUST/CAiRE_in_DialDoc21`. Learning from this work, fine-tuning seems to be critical in improving the text generation performance.

Acknowledgements

I would like to say the greatest thank you to Dae-Hyun Kim, a PhD candidate in Computer Vision at Hanyang University, who inspired and helped me the most during the entire process of coding and designing my experiments. Moreover, I appreciate all the priceless advice from my 2021 cohort members and the faculty at the Centre for Doctoral Training in Interactive AI. I also thank the first author of the Doc2dial paper, Dr Song Feng, and the first author of the CAiRE paper, Yan Xu, for their generous advice that helped me to overcome coding challenges that I faced. Last but not least, I would not have been able to finish this report without my husband's support.

References

Song Feng, Hui Wan, R. Chulaka Gunasekara, Siva Sankalp Patel, Sachindra Joshi, and Luis A. Lastras. 2020. doc2dial: A goal-oriented document-grounded dialogue dataset. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *EMNLP (1)*, pages 8118–8128. Association for Computational Linguistics.

Etsuko Ishii, Yan Xu, Genta Indra Winata, Zhaojiang Lin, Andrea Madotto, Zihan Liu, Peng Xu, and Pascale Fung. 2021. Caire in dialdoc21: Data augmentation for information-seeking dialogue system. *CoRR*, abs/2106.03530.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. cite arxiv:1906.08237Comment: Pretrained models and code are available at https://github.com/zihangdai/xlnet.