MSAI 337: Project proposal

Group 5

Group members: Srik Gorthy, Tianchang Li, Liqian Ma, Shubham Shahi

Project Goals

In the final project, we are more interested in the chatbot problem. This is a hands-on starting point for us to apply language models, text generation, and evaluation techniques.

Specifically, we plan to make attempts in the following aspects. We would test and combine the approaches as needed.

Approach 1

In this approach, we plan to apply a different language model for text generation other than GPT-2. Language models are so important in this task and we would like to try out more powerful ones like BERT, GPT-3, Turing-NLG, MT-NLG. The latter two options are from the latest research outcome in Microsoft Research, which have hundreds of billions of parameters. For this task, it is not necessarily the larger the better. It also depends on the context of the conversations.

Approach 2

In this approach, we plan to encode details such as time, date, day of the week (M-F), section of the day (morning, evening..), phone number, reference number, address, and other integers. Two ways we can go after this:

1) Performing model evaluation on classes of tokens instead of the exact text
2) Build a hierarchical model with a second-layer predicting the exact text in each class

We are aware that the second step could be time-consuming and unfruitful. So we would only attempt it when the major part of the project is done.

Approach 3

We plan to vary the decoding techniques in each version of the model we build, especially top-p sampling and beam search.

Approach 4

We could use both BLEU and ROUGE scores to evaluate each version of the model. The combination of the two scores provides a well-rounded evaluation of both the precision and recall of the model and avoids type I/II error.

First Steps

- Read through the code provided and generate results for the two baseline model
- Implement the above approaches within the time frame as much as possible. Then test each approach along as well as combinations of them.

Preliminary Results

Result table with BLEU scores:

	Greedy	Beam	Top-p
Zero shot	0.081	0.074	No best pred
Language model(50 epochs)	0.39	0.43	0.41
Language model(200 epochs)	0.54	0.61	0.57
Approach 1 (XLNet)	0.56	0.64	0.58
Approach 1 (GPT3)	TBD	TBD	TBD

Approach 1

In this approach, we researched and applied different text generation models like XLNet, T5, GPT3, and RoBERTa(huggingtweets/rockberta). For text generation task, there are basically 2 main types of methods, encoder-decoder and decoder only. T5, XLNet, and RoBERTa are all encoder-decoder methods related to Bert. GPT2 and GPT3 are decoder only models. After comparing the performance of different models and the implementation difficulty, we decided to turn to XLNet and GPT3 as our main language variations.

While our machine can not be finetuned for GPT3 since the OOM error by cuda, we can expect better performance with the GPT3 model. The zero-shot version provides an average 0.13 bleu score. As you can see in the above performance record table, XLNet also has a better performance in bleu score. On average, the fine-tuned XLNet model achieved 0.64 bleu score with beam search. This is what we have expected. XLNet achieved a better score in nearly every natural language task as opposed to GPT2. There are several reasons. First, XLNet uses Byte Pair encoding of a SentencePiece library. Second, it is trained with 146GB data, nearly 4 times that of the GPT2 model. Third, there is a presence of context for each of the words in the sentence for the XLNet model.

Besides, we found different factors impacting the results. First, the number of training epochs can be pivotal to the final performance. Especially for the first 200 epochs, we can see the model performs steadily better. After enough training with given data, the model barely improved and even started to overfit. Second, evaluation logic has to be scrutinized. For metric calculation, we found the “[END]” token can be a negative value for the blue score since all generated sentences do not contain it. Also for beam search, there are two spaces before the

“[END]” token so that the next generated sentence is deprived of the first word. After fixing these bugs in the evaluation code, we would have a more accurate metric score.

Approach 3

As our evaluation score shows, Beam search is an overall powerful decoding method for all our models and shows the relatively best performance across all decoding strategies. Greedy gave the least ideal results and top-p sampling is usually in between greedy encoding and beam search. The way we understand this is that, while the language model is never 100% accurate, greedy decoding only generates the next word by maximum probability. Every word relies on the previous words that have been generated. In this case, greedy decoding tends to aggregate the inaccuracies on each predicted word and amplify the discrepancy between prediction and reference.

Top-p sampling is improved on this by allowing some deviation on the optimal word selection. Here we set p = 0.9, meaning we allow the decoder to deviate among the words that account for the first 90% of the probabilities. This method increases the variance of predictions and decreases the bias at the same time.

Beam search further improves the results by better controlling the balance between the variance and bias of predictions by only allowing deviation among the few best options (here we set 5).

Approach 4

We computed ROUGE scores for each model and each decoding method in parallel (shown below). The same patterns were seen with ROUGE scores. This gave us a more well-rounded understanding of the model performance and confirmed our conclusion: our further fine tuned GPT2 as well as fine tuned XLNet have both beaten the baseline models.

Result table with ROUGE scores:

	Greedy	Beam	Top-p
Zero shot	0.22	0.22	0.22
Language model(50 epochs)	0.36	0.4	0.37
Language model(200 epochs)	0.44	0.49	0.46
Approach 1 (XLNet)	0.47	0.53	0.49
Approach 1 (GPT3)	TBD	TBD	TBD