HW3: Generating Text From Pretrained LMs

Team: Semantic Savants; Jeonghoon Kim, Matthew Olson, and Marco Berriodi

Our team, Semantic Savants, has implemented different decoding techniques to generate text using pre-trained large language models (LLMs) for generation tasks. We have also assessed the generated output text and highlighted the limitations of the current decoding methods. Initially, we utilized the HuggingFace API to generate text using a pre-trained autoregressive language model, GPT2. We used various decoding algorithms, such as greedy search, beam search, top-k sampling, and top-p sampling, based on the prompt "Today, I believe we can finally,". Our team set the maximum length of text as 30 and the number of beams as 5 for beam search. For top-k and top-p, we set k=20 and p=0.95 respectively. We found that a length of 30 gives an effective amount of words to see how the decoding algorithms will perform where it's not too short that all the algorithms seem to be the same and not so long that it keeps on going on a tangent. For top-k and top-p we found that when k = 20 we results that vary enough that it can sound more human. For top p we found that .95 is good at finding words that sound more human-like and not like AI choosing the best probability words every time. For the number of beams, we found that it strikes a good balance between the quality of the generated text and the computational cost since the model considers the top 5 most likely candidates at each step with a beam size of 5, which allows it to explore a range of possibilities while still generating text relatively quickly. The outcomes of generating text using various decoding algorithms based on the prompt "Today, I believe we can finally," are presented in Table 1. We also calculated the perplexity and likelihood of the generated text, shown in Table 2.

Table 2 indicates that the beam search algorithm has the lowest perplexity value among all the decoding methods, followed by greedy search, top-k sampling, and top-p sampling, suggesting that beam search is the most effective algorithm based on perplexity values. However, a negative log-likelihood value implies that the probability of the data given the model is less than 1, which means that the model is not able to explain the data well. Hence, we could assume that the data is noisy or complex.

Decoding Algorithms	Results
Greedy Search	“Today I believe we can finally, and I believe we can finally, get to the point where we can make a difference in the lives of people who are struggling with mental illness.\n”
Beam Search	“Today I believe we can finally, at last, get to the bottom of what's going on in this country.\n\nI think it's important for us all to understand that we”
Top-k Sampling	“Today I believe we can finally, at some point in our lives, start taking responsibility for our own actions. It's been my experience since my teenage years that a good lot of people”
Top-p Sampling	“Today I believe we can finally, with our new platform, reach our full potential.\n\nIt is now your turn to get involved. We are working on the final version. Please”

Table 1. Results of text generation with different decoding algorithm based on the prompt “Today, I believe we can finally,”

Decoding Algorithms	Perplexity	Likelihood
Greedy Search	4.431	-44.66
Beam Search	4.204	-42.08
Top-k Sampling	6.740	-57.24
Top-p Sampling	7.121	-58.89

Table 2. Perplexity and Likelihood of the generated results depending on different decoding algorithms.

Perplexity is an intrinsic method to evaluate the language model. Now, we evaluated the language model with an extrinsic evaluation based on the BART model. Our group selected SAMSum dataset for evaluating decoding algorithms where reference text is provided. We picked “*bart-large-cnn-samsum*” model for abstraction summarization. BART (Bidirectional and Auto-Regressive Transformer) is a pre-trained sequence-to-sequence (seq2seq) model and can be fine-tuned for text summarization. Bart-large-cnn was trained on CNN dataset, which consists of news articles and corresponding summaries. Hence, from the test set of the SAMSum dataset, we simply generated output summary from the first 50 samples. For this task, we used the same four decoding methods with the same parameters, but extended maxLength to 150. We created spreadsheet for the generated text. The example of output shows in Table 3.

test set	greedy_search	beam_search	top_k	top_p
Mary: Are you going by car or train? Tom: Ella rented a car Ella: this makes all of this much faster Mary: good decision	Ella rented a car to go by car instead of the train. Mary and Tom are happy with the decision. Tom and Ella are going by car, not by train, because it makes the journey faster. - Mary is happy with this decision.	Ella rented a car to go by car instead of by train. Mary and Tom are happy with the decision. Tom and Ella are going to meet with Mary soon. They will take the car as it's faster than the train and it makes the journey faster.	Ella rented a car to go by car instead of by train as she wants to get there faster. Mary thinks it's a good decision on her part. Tom thinks it was a good idea on Ella's part as it makes the journey faster.	Ella is going by car as she rented a car. Tom and Mary congratulate her on this decision. - "This makes all of this much faster". Tom and Mary congratulate Ella on this.

Table 3. The example of generated summaries from the test set of the SAMSum dataset were created using various decoding algorithms.

For the automatic evaluation of the generated text, we chose BLEU (content overlap metrics) and BERT score (model-based metrics). The reason is that BLEU (Bilingual Evaluation Understudy) is a widely used metric for evaluating the quality of machine translation outputs. It measures the n-gram overlap between the generated summary and the reference summary [1]. We thought that BLEU could be a good choice for the SAMSum dataset because it contains messenger-like conversations with summaries, and BLEU is commonly used to evaluate the quality of summaries and translations. Moreover, the BERT score is a metric that measures the similarity between the generated summary and the reference summary using contextualized word embeddings [2], so we thought that the BERT score is a good choice for the SAMSum dataset because it captures the nuances of language use in messenger-like conversations, which may contain informal language, slang words, and typos. Therefore, we hypothesized that using BLEU and BERT scores together can provide a comprehensive evaluation of the quality of the generated summaries for the SAMSum dataset. By calling those evaluation functions, we could simply evaluate the generated summaries, and the average scores for BLEU and BERT score are shown in Tables 4 and 5. After analyzing both scores, we found that some of the BLEU scores are zero, which implies that there is a perfect mismatch between the generation output and test summary. When searching for the reason [3], we found a potential issue. One potential issue with using BLEU to compare the performance of different algorithms for text generation is that it assumes that the generated text should be similar to a set of human-generated reference texts.

This assumption may not always hold when different algorithms are used because they may generate text that differs significantly from the human reference texts (summary of test set). For BERT score, it showed a very high score because it captures the nuances of language use in messenger-like conversations, which may contain informal language, slang words, and typos. Furthermore, we used F1 instead of prediction or recall. The reason is that the F1 score is a weighted harmonic mean of precision and recall, which provides a balance between the two metrics. In the case of the SAMSum dataset, precision and recall are both important in evaluating the quality of the generated summaries, as they both capture different aspects of the semantic similarity between the generated summary and the reference summary.

Average of BERT score (F1)	
Greedy search	90.22%
Beam search	90.10%
Top-k sampling	90.44%
Top-p sampling	90.71%

Table 4. The average of BERT score for each decoding algorithms for 50 samples

Average of BLEU score	
Greedy search	10.31%
Beam search	9.19%
Top-k sampling	11.16%
Top-p sampling	12.49%

Table 5. The average of BLEU score for each decoding algorithms for 50 samples

After automatic evaluation, we performed human evaluations for the first 20 samples for each algorithm. Each group member manually annotates the samples' scores with Likert scale (1-5). For each group member evaluation, we had a separate google sheet so one person's score would not influence by anthers. For the human evaluation metrics, we chose to do factuality and fluency. This is because we concluded that summarizing the fact and word choice should be considered the most for text summarization and the factuality of the summary is very important due to that is what a summary is and how fluent the summary has an impact on how well it can be read.

For the overall score of the algorithm, we took the average voting of each algorithm for each prompt. Then we took the average of all the prompts to get the average Factulity or fluency of that algorithm. We did average due to us having a small pool of human evaluators. The difference between human and automatic evaluation is very big with the BERT and bleu being very drastically different with BERT scoring very high while bleu very low. This is very different

than the human scores which seem to be not very different from each other. The factuality score is around 2.6-3.4 and the fluency score is between 2.9-3.3. With the human evaluation score, we can see greedy and beam have a lower average score than top-k and top-p which may indicate that sampling the top percentile instead of choosing the best one leads to more human-like text and summaries.

Average Evaluation Scores of First 20 Samples				
	Automatic Bert	Automatic Bleu	Human Factuality	Human Fluency
Greedy Search	90.22%	10.31%	2.97	2.88
Beam Search	90.10%	9.19%	2.67	2.87
Top-k sampling	90.44%	11.16%	2.92	3.18
Top-p Sampling	90.71%	12.49%	3.38	3.3

Table 6. The Average Evaluation Scores of first 20 samples with Automatic Evaluation being a percentage and Human Evaluation being between 1 and 5

Upon conducting both evaluations, we discovered that identifying a suitable metric for evaluation can be challenging since certain metrics perform better in particular situations. Additionally, we observed that automatic evaluation is more rapid but may lack accuracy in comprehending details, whereas human evaluation is a slower process and may lead to inconsistencies in the evaluation process.

References

[1]“Metric: bleu” HuggingFace. [Online]. Available: https://huggingface.co/spaces/evaluate-metric/bleu. [Accessed: Mar. 23, 2023]

[2]“Metric: bert_score” HuggingFace. [Online]. Available: https://huggingface.co/spaces/evaluate-metric/bertscore. [Accessed: Mar. 23, 2023]

[3]OpenAI, “ChatGPT,” chat.openai.com. [Online]. Available: https://chat.openai.com/