IA-327:
Large Scale Generative Models for NLP and Speech Processing

Lab 3

Student:

Benjamin TERNOT

19/02/2025

1 Questions

Q1: What do the "dependency" metrics measure?
The "dependency" metrics measure the accuracy of the predicted dependency links in the parse trees compared to the gold (reference) parse trees. They are computed against the sets of dependency links extracted from both the predicted and gold parse trees.

Q2: What is BM25? How does this retriever work? BM25 is a ranking function used by search engines to estimate the relevance of documents to a given search query. It is based on the probabilistic retrieval framework and considers term frequency or document length.
The FewShotRetriever class uses the BM25 algorithm to retrieve the most relevant examples from the training set for a given query. It works as follows:

- Initialization:
  The class is initialized with a dataset containing items from the training set.
- Index Building:
  The ‘build_index method‘ creates an index of the corpus using the BM25 algorithm.
- Sample Retrieval:
  ‘The get_samples‘ method takes a query, tokenizes it, and retrieves the top n most relevant examples from the indexed corpus using the BM25 algorithm.

2 Result comments

2.1 Performance with Respect to Number of Few-Shot Examples

See Figure 1 for the graph associated.

- **SmolLM-135M**: The dependency precision improves as the number of few-shot examples increases, peaking at 2 few-shot examples and then slightly decreasing. The subtree precision follows a similar trend but with lower values.
- **distilgpt2**: This model shows a consistent increase in dependency precision with the number of few-shot examples, reaching its highest precision at 4 few-shot examples. Subtree precision also increases but remains relatively low.
- **gpt-neo-125M**: The dependency precision increases with the number of few-shot examples, peaking at 4 few-shot examples. Subtree precision shows a slight increase but remains low overall.

2.2 Amount of Computation Needed (FLOPs per Sentence)

See Figure 2 for the graph associated.

- **SmolLM-135M**: The FLOPs per sentence increase with the number of few-shot examples, indicating higher computational requirements as more examples are used.
- **distilgpt2**: This model also shows an increase in FLOPs with the number of few-shot examples, with a significant jump at 4 few-shot examples.

- **gpt-neo-125M**: The FLOPs per sentence increase with the number of few-shot examples, with the highest computational cost observed at 4 few-shot examples.

2.3 Comments

- **SmolLM-135M**: This model shows a good balance between performance and computational cost, with a noticeable improvement in precision with a small number of few-shot examples.
- **distilgpt2**: This model demonstrates a steady improvement in precision with more few-shot examples but at a higher computational cost, especially at 4 few-shot examples.
- **gpt-neo-125M**: While this model shows an increase in precision with more few-shot examples, the overall precision remains lower compared to the other models, and the computational cost is relatively high.

Overall, **SmolLM-135M** appears to be the most efficient model in terms of balancing performance and computational cost, while **distilgpt2** shows the best improvement in precision with more few-shot examples but at a higher computational expense. **gpt-neo-125M** has the lowest precision and highest computational cost among the three models.

It is also important to note that with too many few-shot examples, the output becomes too long, leading to a significant drop in precision due to model limitations on token length.
Also, with no few shot example, all model don't perform well at all.

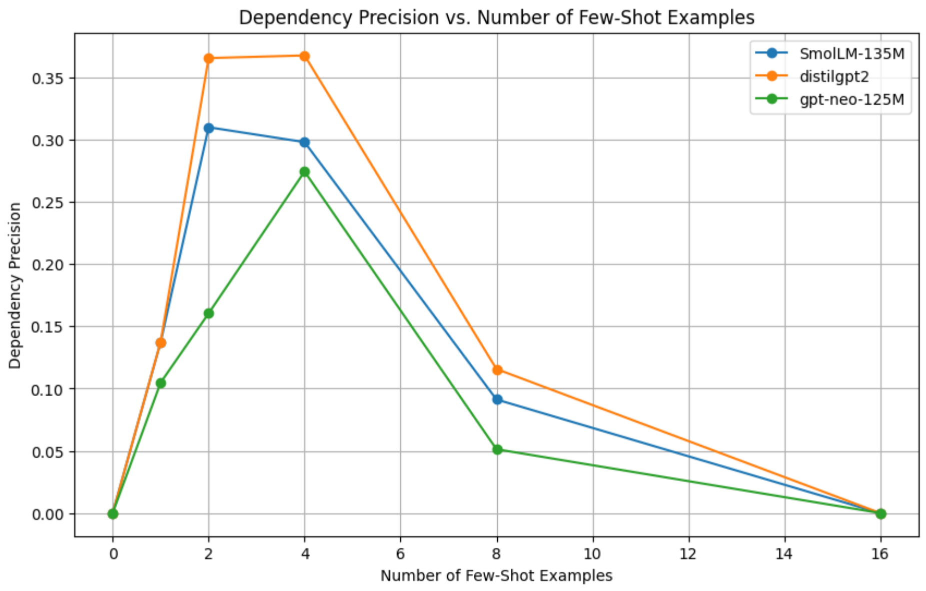

Figure 1: Comparison of precision with respect to the number of Few-Shot

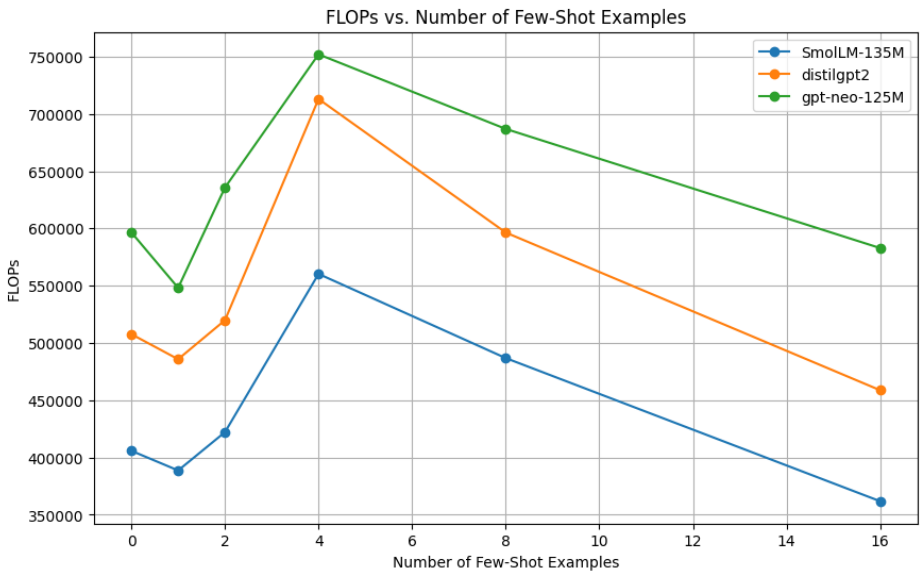

Figure 2: Comparison of FLOPs with respect to the number of Few-Shot