Lab Note: Finetuned Block LLM

Team name: The Query Lab
Members: Mark Henri Mörsheim, Maryam El Ghadioui

After removing bot users, another group identified and clustered three types of real users. One of them were users, who used a moderate amount of filters. Simulating these users was our goal with this submission. Here's an example of what this type of user may use as a query:

"climate change" AND documentType:"research" AND fieldsOfStudy:"environmental science" AND (yearPublished>=1999 AND yearPublished<=2006)

We used an LLM approach to contrast our first submission, and because this type of query seems to be fitting for an LLM, but because of the private nature of our real user data, using an external service was not an option. Luckily, fine-tuned smaller (local) LLMs can often compete or even outperform large generic models for very specific tasks, obviously depending on the task and training data. This is a great way to balance the inferior hardware at hand.

Training and generation were done on an **AMD RX 7800XT (*16GB of Vram*)** on **Arch Linux** / **Ubuntu 24.04.02**, using **ROCm 6.4** and the **PyTorch Nightly** version to match the ROCm version. We used Meta's **Llama3.2 (3B)** as our base model, because it's relatively new, very small in size, yet powerful (enough).

In preparation for the training, the cluster of queries had to be converted to a "question - answer" structure. The mentioned cluster contained *618654* queries, exceeding the limits of our hardware setup. Considering the relative success of the block-training, we settled on *3000* "question - answer" pairs, where we simply used "Output:" as the 'question' every time. Thus, a *Keyphrase* was introduced in training, which the model recognizes before generating an answer that is based on the LoRA training. This is the reason our prompt is structured like this:

prompt: str = f"{query}. {keyphrase}"

 # Example Prompt:

climate change misinforatmion wildfires. Output:

Despite the small amount of training data, the results were surprisingly convincing. The model understands the prompts and extended it in meaningful ways. When it comes to the filters, it mostly only added a yearPublished-filter at the end of the query like that:

...(yearPublished>=1975 AND yearPublished<=2025)

It seemed to have learned that the upper end of that yearPublished-filter is the current year for most queries, either 2025 or 2024. This will obviously become outdated. On the other hand it's hard to tell how it's determining the lower end of that filter: It maybe be partly arbitrary and partly related to the topics at question.

As mentioned before, the model is small, yet already powerful, and adds meaningful terms, articulating a great understanding of the prompt and its topics. The real gains will come from better fine-tuning and more training data.