**Fine-Tuning and Context Injection**

**Fine-tuning is an approach where a pre-trained LLM is retrained with new data.** This process can help optimize the LLM for specific tasks such as sentiment analysis and improved prediction skills. When retraining the model, the weights and biases of its neural network are adjusted based on training with the new data. The downside of fine-tuning is that it can be expensive, time-consuming, and requires a significant amount of computing power to retrain the model.

**Context injection (also known as in-context learning) is an approach that is focused on utilizing the prompt that is fed into the LLM.** Rather than modifying the LLM, the relevant context is injected into the prompt. This allows the LLM to answer questions by utilizing the context provided in the prompt.

Although each method has its unique applications and advantages, as a beginner, the context injection approach would be the preferred method. In this article, we will focus on a project involving context injection.

**General Steps of Context Injection**

Screens screenshot of a chat

Description automatically generated

Image Credit: Dominik Polzer

The diagram above is an excellent introduction to the basics of context injection. We will summarize the process of context injection as follows:

1. Collect the structured or unstructured data that you would like the LLM to answer questions about.
2. Process this data by loading the data (text loader) and tokenizing it (text splitter). Tokens are essentially a short string of characters, generally 4 characters in length. For example, the word “generative” may be split into tokens as “ge”, “n”, “erat”, and “ive”. The LLM processes data as tokens.
3. Tokens will be fed into an embedding model which converts tokens into embeddings. Embeddings are how words and sentences are represented in a vector space. (More in-depth explanation in the next section)
4. Vectors produced from the embedding model are stored in vector databases. In our case, we will be using ChromaDB.
5. Now for the interesting part. When a user asks a question, we will convert their query into a vector and search for the nearest vectors from the database. Essentially, this process will locate the most relevant text chunks to the user’s query and convert them back into text.
6. The user’s question and the relevant text chunks (context) will be included in a prompt template that is given to the LLM. Without any modification to the original LLM, the model can provide impressive answers to the query using the injected context.