What is Fine-Tuning of an LLM?

We will examine fine-tuning an LLM or large language model. This involves customizing an LLM for particular use cases or requirements. However, before discussing fine-tuning in-depth, it's important to note that there are other ways to customize an LLM.

Here is a brief look at some of them.

I. Prompt Engineering and Template Design.
To get better results from an LLM, sophisticated prompting techniques may be used. This may include adding a data set or examples. Platforms like LaneChain allow for the creation of prompt templates for this purpose.

II. Retrieval, Augmented Generation, or RAG.
This is where you add data sets to the LLM. They can be fairly large. With RAG, you'll query this information, usually from a vector database. This is a specialized database that has the vector embeddings for the data sets. The query of the information will often use advanced search capabilities like semantic search.

III. Fine-tuning
This is a much more sophisticated approach than prompt templates or RAG. Here are some of the benefits.

- Domain-specific knowledge. You can tailor the LLM for a particular purpose. This could be for your sales department, or customer support, or legal, or so on. For the most part, you can tailor it to the jargon, the specialized rules and requirements of your organization. This also helps to boost the accuracy of the responses.
- Hallucinations. A hallucination is a false or misleading response from an LLM. How does this happen? Well, there are a couple reasons. One is that the LLM is fairly general, and the other reason is that the underlying model, called the transformer model, is based on probabilities. However, with a fine-tuned LLM, you should be able to mitigate the hallucinations.

Now let's take a look at the process of fine-tuning. You'll select a data set for a task, domain, or category. For example, if you're building a customized LLM for human resources, you would include documents such as for HR policies and procedures, job descriptions and listings, employee communications, and so on. Basically, you will be using the kinds of data that will not have been a part of the initial LLM. For the most part, the data sets need to be large, say thousands or tens of thousands of examples. This will help with the learning process, and of course, it helps to have clean data. For example, you'll try to avoid such things as missing information. The data is usually in a JSON format. It will have a prompt completion pair.

Swamy Sivasubramaniyan
Swamy.Sivasubramaniyan@USVenturesLTD.com

```
[
   {
      "prompt": "How should an HR professional respond when an employee asks about maternity leave policies?",
      "completion": "As an HR professional, it's important to provide clear and supportive information. You can respond with, 'Our maternity leave policy offers 12 weeks of paid leave. I can email you the detailed policy document and answer any specific questions you might have.'"
   },
   {
      "prompt": "What is the best way to handle a conflict between two team members?",
      "completion": "In handling conflicts, it's important to remain neutral and understanding. A good approach is to arrange a meeting with both parties to discuss the issue openly and calmly, and to facilitate a solution that acknowledges and addresses both parties' concerns."
   },
…
```

Here's an example. So, for the first element of this JSON structure, we have a question from an employee that asks about maternity leave policies, and then we add the completion, and here we get the answer to it, and it's based on the underlying policies of the organization. So by having many of these types of examples, your fine-tuned LLM should have much better responses.

The next step in the process is that you'll train the data for the LLM. First you'll divide the data set. There is the training data. This is what the model will learn from. Then you'll have the validation data set to test and evaluate the results. So with the fine-tuning process, you'll be adjusting the parameters to get the best results. The parameters are the weights and the biases. These are values to find the relationships among the tokens. The token is a word, part of a word, or a character. One path through the process is called an epoch, but you'll usually need to have several epochs to improve on the model. Essentially what's happening is that some of the general understanding of the initial LLM is being overwritten with the training of the new data set. Then how is fine-tuning different from prompt templates and RAG? Basically, prompt templates and *RAG do not adjust the underlying parameters*. The initial model remains the same. This is why fine-tuning can mean much better outcomes from an LLM, but there are certainly some *drawbacks* to consider.

Complexity. It is far from easy to get the right data sets and do the fine-tuning. Basically, you need people who have specialized data science skills.

Cost. It can be expensive to do the fine-tuning. You'll likely need powerful systems like GPUs or graphics processing units. There are also the costs of hosting the model. OK then, we're now finished with this video. In it, we've looked at fine-tuning of LLMs.

Swamy Sivasubramaniyan
Swamy.Sivasubramaniyan@USVenturesLTD.com