Dataset: https://huggingface.co/datasets/kalinkov/tailwindcss_components
LoRa adapter: https://huggingface.co/kalinkov/phi3_lora_model
GGUF model: https://huggingface.co/kalinkov/Phi3_tailwindcss
Google Drive:
https://drive.google.com/drive/folders/1qV80Wh-8ytMG1uOTKBn4uPq4b_aXb-gE?usp=share_link

Fine-tuning a language model on a custom dataset, such as a collection of Tailwind CSS components, can significantly enhance its utility for specific tasks. The Phi-3 Medium 4K Instruct model, designed for natural language understanding and generation, can be adapted to generate, modify, and understand Tailwind CSS code more effectively after such fine-tuning. This process typically involves several key steps, from data preparation to model training and deployment.

1.Conceptual Overview

The primary goal of fine-tuning a language model like Phi-3 Medium 4K Instruct on a dataset of Tailwind CSS components is to enhance its ability to understand, generate, and manipulate HTML and Tailwind CSS code. Tailwind CSS is a popular utility-first CSS framework that allows developers to apply styles directly in HTML using predefined classes. Fine-tuning the model with a specialised dataset helps the model become more proficient in handling the specific syntax, patterns, and conventions used in Tailwind CSS, thereby making it a valuable tool for developers.

2.Data Preparation

The first step in this process is data preparation. The dataset consists of various Tailwind CSS components, each with its unique structure and class combinations. This dataset was formatted to match the Phi-3 chat template and uploaded to Hugging Face, a popular platform for hosting and sharing machine learning models and datasets.

3.Fine-Tuning Process

With the data prepared, the next step is to use a framework like Unsloth for the fine-tuning process. Unsloth simplifies the task of fine-tuning large language models by providing tools for data preprocessing, training, and evaluation. The Phi-3 Medium 4K Instruct model is loaded and fine-tuned on the custom dataset of Tailwind CSS components. During this phase, the model's parameters are adjusted to better predict and generate Tailwind CSS code based on the training data.

3.1Choice of Unsloth

Unsloth was chosen as the framework for fine-tuning the model due to several key advantages:

1. Ease of Use: Unsloth provides a streamlined interface for setting up and managing the fine-tuning process. Its user-friendly design reduces the complexity typically associated with fine-tuning large language models.

2. Efficient Data Handling: Unsloth includes robust tools for data preprocessing, which ensure that the dataset is optimally prepared for training. This includes handling large datasets efficiently, which is crucial when working with extensive collections of code snippets.

3. Customizable Training Parameters: Unsloth allows for fine-grained control over training parameters, enabling customization to fit the specific needs of the Tailwind CSS dataset. Parameters such as learning rate, batch size, and number of epochs can be easily adjusted to optimise model performance.

4. Integration with Hugging Face: Since the dataset was hosted on Hugging Face, Unsloth's seamless integration with this platform made it easier to load the dataset and manage the fine-tuning process. This integration ensures smooth data flow and model management.

3.2Training the Model

The actual fine-tuning involved several steps:

1. Loading the Model and Dataset: The Phi-3 Medium 4K Instruct model was loaded into the Unsloth framework, along with the custom Tailwind CSS dataset.

2. Preprocessing: The dataset underwent preprocessing to convert the Tailwind CSS components into a format suitable for the model. This included tokenization, where the text data (CSS classes and HTML) was transformed into tokens that the model can process.

3. Fine-Tuning: The fine-tuning process involved adjusting the model's weights based on the Tailwind CSS data. During the training, the model's predictions were compared against the actual data, and adjustments were made to minimise errors.

4.Practical Applications

The fine-tuned Phi-3 Medium 4K Instruct model can now assist developers by generating HTML/Tailwind CSS websites while using the learned components. This can significantly speed up the development process and ensure consistency in the application of Tailwind CSS conventions. Additionally, such a model can be integrated into various development tools and platforms, providing real-time assistance and enhancing the overall productivity of developers working with HTML and Tailwind CSS.

By following these steps, the process of fine-tuning the Phi-3 Medium 4K Instruct model on a custom dataset of Tailwind CSS components demonstrates a practical application of machine learning to streamline and enhance web development workflows.