****

**TinyLlama-1.1B-Chat-Version**

**A Technical Review for AAIE Text Detection**

TinyLlama is a compact, 1.1 billion-parameter language model designed which was pre-trained on approximately 3 trillion tokens. This built on the architecture and tokenizer of [Llama 2](https://arxiv.org/pdf/2307.09288.pdf). The chat version is specifically fine-tuned for dialogue which can makes it so effective for our task by learning to master natural, human-like conversation, it inherently became an expert at recognizing the artificial patterns that give AI-generated text away.

* **Developer/Organisation**: This project is currently contributed by [Peiyuan Zhang](https://veiled-texture-20c.notion.site/Peiyuan-Zhang-ab24b48621c9491db767a76df860873a?pvs=4) , [Guangtao Zeng](https://github.com/ChaosCodes) , [Tianduo Wang](https://github.com/TianduoWang) and [Wei Lu](https://istd.sutd.edu.sg/people/faculty/lu-wei/) from the StatNLP Research Group of Singapore University of Technology and Design.
* **Key Features**:
  + **Model Size**: 1.1 billion parameters.
  + **Context Window**: 2048 tokens.
  + **Architecture Type**: Same as the base model (Llama 2 architecture) but fine-tuned for dialogue.
  + **Target Use Case**: Conversational AI, chatbots, and interactive applications that require instruction-following capabilities.
  + **Training Data**: Built upon the base model and then fine-tuned on conversational datasets.
  + **License**: **Apache 2.0 License**.
  + **Link to Model**: <https://github.com/jzhang38/TinyLlama>

**Compute Requirements**

Fine-tuning the TinyLlama model is designed to be highly accessible and does not require extensive computational resources. We can accomplish this task using a single consumer-grade GPU, such as a Google Colab T4, an NVIDIA RTX 3060 (12GB), or an RTX 4060 (8GB). For memory, the process requires approximately 8 GB of VRAM when using efficient techniques like QLoRA, although having 16 GB is recommended for a smoother experience. Additionally, our system should have at least 16 GB of standard RAM to handle the dataset and data loading without issues. The storage footprint is also minimal, with 5 GB to 10 GB of space being sufficient for the model files, dataset, software, and training checkpoints. In terms of speed, while the fine-tuning process on a dataset of a few thousand examples can often be completed in under an hour, it's important to remember that this ultimately depends on how much data we are fine-tuning and the specific computational power you have available.

**Pros & Cons**

* Advantages: We don't need a supercomputer to work with it; it can be fine-tuned and run on the kind of hardware that can be find in a typical computer lab or even on a student's laptop, which is perfect for an academic project. This accessibility, combined with its fast-processing speed, makes it great for systems that need to check a lot of documents quickly. Because it's built on the same foundation as the popular Llama 2, it plugs right into a whole ecosystem of ready-made tools, which really simplifies the development process. Plus, since the base model has already been trained on a massive amount of text, it has a solid grasp of language, so you only need to give it a relatively small amount of your own data to teach it the specific task.
* Disadvantages: The model can only look at a couple of thousand words at a time, so if you need to analyse a long essay, you'll have to chop it into smaller pieces first. Also, because it's a smaller model, it doesn't have the deep reasoning power of its much larger cousins. This might mean it could struggle to catch more advanced AI-generated text that is skilfully designed to sound human. Finally, to get it to run so efficiently, the model is often compressed, and while this works wonders for saving space, it can sometimes slightly reduce its overall accuracy.

**Model Selection Recommendation for the AAIE Project**

As we are in the initial phase of this project, our model selection strategy must be flexible and pragmatic, primarily driven by two key factors: the volume of data we will have for fine-tuning and the local computing power at our disposal. To make the most of our available hardware, we can adjust training parameters; for instance, utilizing local machines with less memory is feasible by reducing the batch size, though this will increase the overall computation time.

Our recommendation is to start with the TinyLlama-Chat series. Although these are compact models, they are built upon a foundation of extensive pre-training on a massive language dataset, making them highly effective for specialized tasks once fine-tuned.

Within the chat series, we have three versions (V0.1, V0.3, and V0.4). The choice among them presents a classic trade-off. The more advanced versions, trained on more data, are likely to yield higher accuracy but will demand more time and computational resources for fine-tuning. Conversely, starting with the smallest version might not deliver the quality of results we need.

Therefore, I recommend an empirical approach to make our final selection. Once the training data becomes available, we can move into a short-term experimental phase to fine-tune and evaluate all three chat versions. This will allow us to make a data-driven decision, ensuring we select the model that offers the best balance of performance and efficiency for our specific needs and constraints.