Adapting & Aligning Large Language Models to Targeted Code-Gen tasks.

Teodor Poncu[1,2], Vlad Florea[1,2]
1 - University Politehnica of Bucharest, RO. 2 - Unakin.ai, UK.

Problem Setting

Specialized programming tasks, such as game development, often suffer from a scarcity of available online data and well-trained developers. This is particularly relevant when training LLMs, for two key reasons:

- **Scarcity**: Given the lack of accessible online data, unsupervised training is challenging due to the limited number of available training samples and out of distribution characteristics.
- **Cost**: High-quality instruction-output pairs are costly and require scarcely available specialists that can generate this data.

For our experimentation, we assemble several datasets:

1. A training set of roughly **1000 files** extracted from permissively licensed Unity projects hosted on GitHub.
2. A test set that emulates a real-world scenario where we want to adapt to a production-grade codebase. To simulate this, we procure projects listed on the Unity Asset Store, collecting about **600 high-quality files**.
3. An alignment assessment test set, for which we collect approximately **100 hiqh-quality instruction pairs**.

Approach

We experiment on the **Llama-v2** [1] and **Code-Llama** [2] model variants. We aim to gauge the sensitivity of these models to a potential lack of specialization in the underlying language model. We utilise perplexity as a metric to benchmark our approach.

Our primary finding underscores the critical role of **unsupervised pre-training** in equipping the model with domain-specific knowledge. The key insights from our experimentation can be summarized as follows:

- **Leveraging chat models** as a foundational model offer a free lunch in terms of instruction-following capabilities when paired exclusively with **unsupervised pre-training**.
  - The usage of a static next-token prediction prompt hinders both alignment and overall model perplexity. We hypothesize that the static instruction works as a non-optimal tunable prefix during training.
- **Not using a highly specialized base model does not lead to catastrophic results**. We successfully narrow the performance gap between the original **Llama-v2** and **Code-Llama** models to a substantial degree by employing a careful online training schedule (**Fig. 1**).

```
if micro_batch_loss > malformed_threshold:
  continue
if micro_batch_loss > spike_threshold:
  micro_batch_loss = scale_spike(micro_batch_loss)
microbatch_loss.backward()
accumulated_micro_batches += 1

if accumulated_micro_batches == n_micro_batches:
  opt.step()
```

Figure 1: Python-like pseudo-code for loss-scaling during training

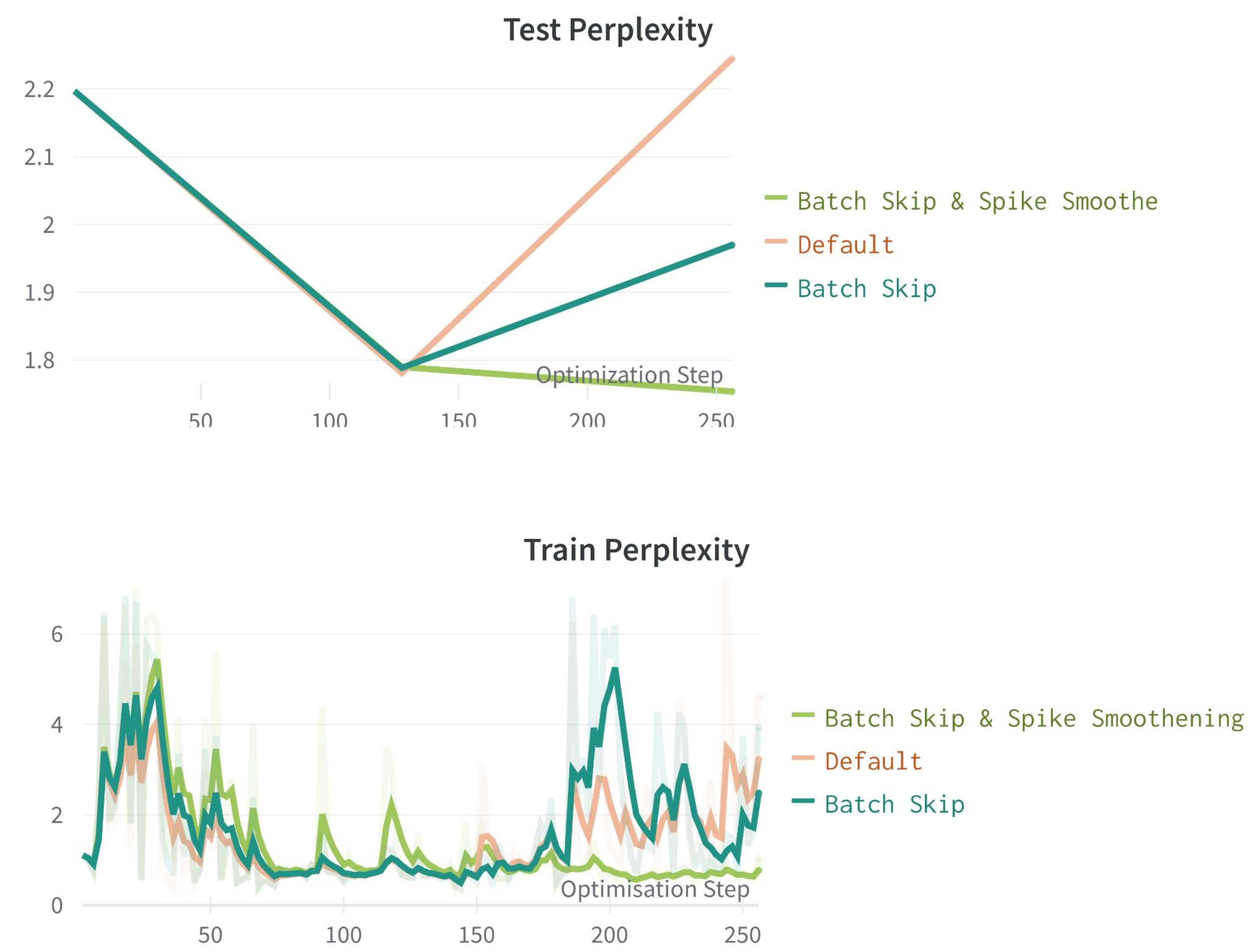

Figure 2: Showcase of non-specialized model loss spikes under certain token sequences. We use a fully deterministic sampling procedure across all experiments.

We compare the effectiveness of full fine-tuning and parameter efficient fine-tuning. In this limited data regime we observe minor performance improvements when using PeFT like techniques.

We improve the computational efficiency **(eq. 2)** of all LoRA-like [3] methods **(eq. 1)**. Furthermore, we note that setting a higher rank using a high dropout **(i.e. > 0.5)** yields improved perplexities. Concurrent works such as DyLoRa [4] showcase similar results.

$$\mathbf{H} = \mathbf{W}^\mathbf{T}\mathbf{X} + (\mathbf{AB})^\mathbf{T}\,\mathrm{Dropout}(\mathbf{X}) \qquad (1)$$

$$\mathbf{H} = (\mathrm{Dropout}(\mathbf{AB})^\mathbf{T} + \mathbf{W}^\mathbf{T})\,\mathbf{X} \qquad (2)$$

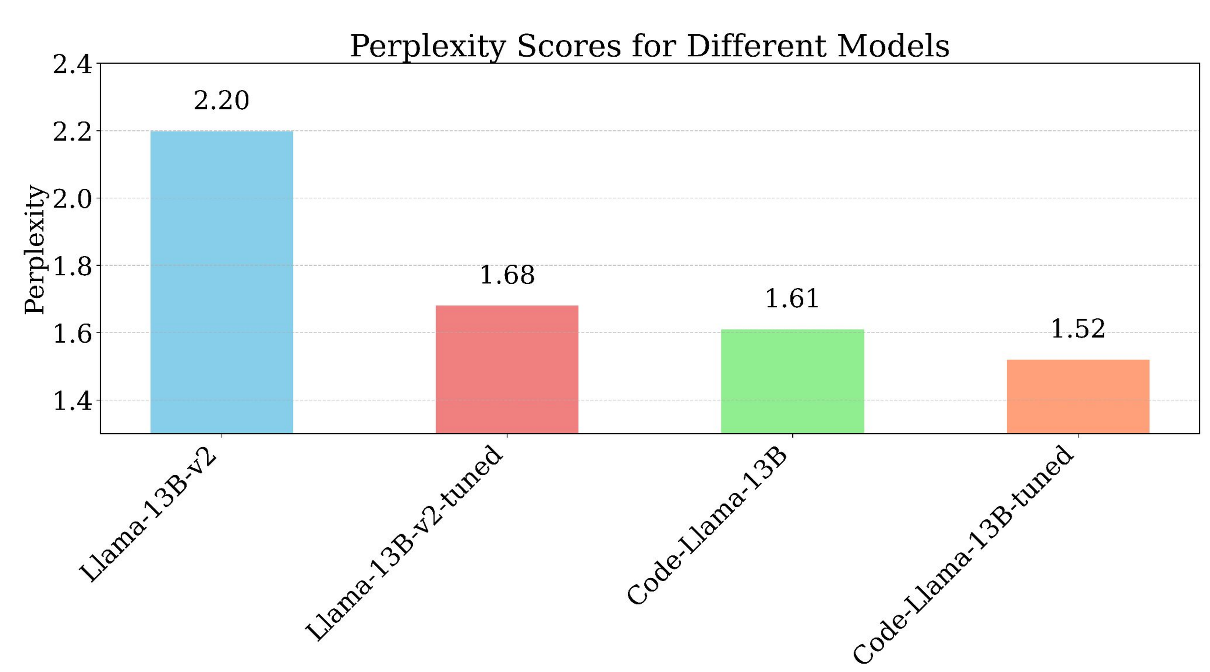

Figure 3: Quantitative comparison between various model families.

Alignment & Self-Instruction

We experiment with several methods of generating instruction fine-tuning using synthetic data [5, 6, 7]. We conducted experiments on two types of instruction and our key findings are:

- **LLMs are not yet perfect encoder-decoders.**
  - We test several description and summarization seed prompts. Models cannot fully infer the original file content from their own descriptions. We also note that a **2x drop** in description length leads to an almost **10x decrease** in the model's ability to regenerate a script similar to the original one.
- **Model scale is important when generating instructions.** We rely on instruction back-translation [7] to generate synthetic input-output pairs. We use the 13B and 34B models both as generators and rankers.
  - The smaller model is consistently ranked lower than the larger model, with an average rating difference of **1.2** points (on a 1 to 5 scale).
  - Self-rating leads to **"fake-agreement"**. When using the same model as both generator and discriminator we notice that we get an approximate score of **5** (on a 1 to 5 scale).
- **Humans provide poorer alignment signals on average.** We rank the human annotated instructions with the 13B and 34B models, as well as with the 13B fine-tuned models.
  - The human annotators get an average score of **1.3**, which is about **2x lower** than the lowest performing model. A common explanation that the model provides is a **lack of sufficient clarity and details in the instruction.**

Conclusion

We find that we can **adapt Instruction Following LLMs to highly specialised tasks without using instructions** but solely through careful unsupervised pre-training.

The empirical evidence showcasing the **effectiveness of online loss scaling** suggests that general pre-training efficiency might benefit from better **data or task curriculums**. Given LoRA computes **pseudo-gradients** for the weight matrices, **high-dropout rates loosely mimic gradient averaging**. We believe meta-learning techniques might improve LLM training.

Self-instruction techniques, although showcasing great performance in general language modelling tasks, **proved lacklustre for the task of complex code generation**.

References

[1] Touvron, Hugo, et al. "Llama 2: Open foundation and fine-tuned chat models." arXiv preprint arXiv:2307.09288 (2023).
[2] Rozière, Baptiste, et al. "Code Llama: Open Foundation Models for Code." arXiv preprint arXiv:2308.12950 (2023).
[3] Hu, Edward J., et al. "Lora: Low-rank adaptation of large language models." arXiv preprint arXiv:2106.09685 (2021).
[4] Valipour, Mojtaba, et al. "Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation." arXiv preprint arXiv:2210.07558 (2022).
[5] Taori, Rohan, et al. "Alpaca: A strong, replicable instruction-following model." Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html 3.6 (2023): 7.
[6] Li, Xian, et al. "Self-Alignment with Instruction Backtranslation." arXiv preprint arXiv:2308.06259 (2023).
[7]Lee, Harrison, et al. "RLAIF: Scaling Reinforcement Learning from Human Feedback with AI Feedback." arXiv preprint arXiv:2309.00267 (2023).