

Team InnovAdapt: Training LoRA for Finance Tasks

Charles Kniffin

ckniffin6@gatech.edu

Winston Lai

wlai46@gatech.edu

Yohann Venkateswaran

ysv3@gatech.edu

Abstract

With the release of ChatGPT in November 2022 LLMs have seen an explosive growth in popularity. Frontier models such as Claude 3 Opus have shown remarkable abilities in tasks related to math, logic and summarizing however they require enormous resources to train. Companies such as Mistral have made smaller 7 billion (7b) parameter class models publicly available however these models still need resource intensive fine tuning on specific tasks to achieve frontier performance. In this paper we examine a class of Parameter Efficient Tuning (PEFT) methods known as Low Rank Adapters (LoRA) as a means to further tailor the Mistral 7b model to the specific task of finance. We examined the trade offs in different LoRA methods while attempting to train an adapter to outperform a Claude on finance related tasks. We show the training times, hyper parameters and the testing results for a variety of configuration. We found that in spite of what some organizations claim about the capabilities of adapters achieving frontier level performance is not straight forward and the the PEFT landscape is still it's infancy when it comes to practical application.

1. Introduction/Background/Motivation

1.1. Project Overview

We attempted to train what are known as Low Rank Adapters (LoRA) to tailor an open source LLM specifically for finance related tasks. Using a LoRA consists of training a small network parallel to the original model which has all of its trainable parameters frozen. A typical LoRA network can have less than 0.1% as many parameters as the original model allowing for a fraction of the computation to be used relative to a fine tune of all model parameters. Thus we focus on LoRA as it has the potential to solve the problem of needing massive computational power and time for a full fine tune while potentially yielding similar results. We select the Mistral-7b base model (as opposed to an instruction tuned model) as the base model to minimize the bias of the starting model and emphasize the role of the adapter. We evaluate the models primarily on a set of hand crafted questions to try and replicate real world use. As part of

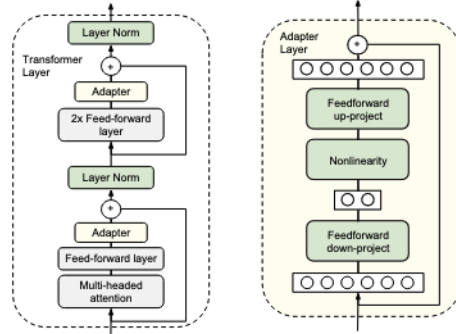


Figure 1. A traditional adapter as proposed by Houlisby et al. [2019]

solving this problem we examine 3 LoRA frameworks and compare their performance against frontier and other open source models. Succinctly our objective was to train a variety of LoRA adapters using the LoRA, QLoRA and LoftQ frameworks to try and match the performance of a frontier model on finance related tasks by benchmarking with hand crafted questions.

1.2. Prior art and Project Impact

Adapters were first proposed in 2019 by Houlisby et al. [2019]. This subsequently leads to LoRA in 2021 proposed by Hu et al. [2021]. Traditional adapters required adding additional layer directly into the existing model as shown in Figure 1. This increases inference latency and requires the ability to directly manipulate the original model. As shown in Figure 2 LoRA creates a network that works in parallel to the original model. This effectively shifts the bias of the model due to the output of the model becoming a combination of the base network and the adapter network. Training is then conducted on labeled data by applying a query to both the base and LoRA network and comparing the result to the label. Loss is then calculated and only back-propagated on the LoRA network as the base network has had it's parameters frozen.

A major follow on to standard LoRA was QLoRA proposed in 2023 by Dettmers et al. [2023]. QLoRA first quantizes the base model, converting the high precision parameters (typically 32 or 16bit) into lower precision parameters

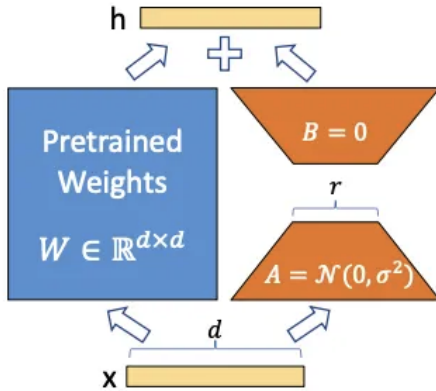


Figure 2. LoRA as proposed by Hu et al. [2021]

(such as 8 or 4 bit). This significantly reduces the amount of GPU memory required for training while not significantly reducing performance. The crux of QLoRA is that the full precision of the base model is often not needed to represent what the model has learned. In other words using the 65536 different bins in a 16bit model does not result in greater accuracy over the 16 bins in a 4bit model. We ask readers to refer to the QLoRA paper for further treatment of this.

Most recently, LoftQ (Li et al. [2023]) is introduced as the newest LoRA implementation performing better than both LoRA and QLoRA at very low bits. LoftQ combines the ideas from LoRA and QLoRA and utilize both low rank approximation and quantization to approximate the pre-trained weight matrices resulting in high accuracy and better performing models. The motivation of LoftQ stems from the QLoRA initialization, where models are zero-initialized to quantized pre-trained model. It is undesirable to randomly initialize models as in each forward pass random noises are added to the adapter during training which could lead to instabilities. However, the effect of precision approximation due to quantization introduces discrepancy in accuracy and can adversely impact the initialization of fine-tuning, especially for low-bit situations. LoftQ was developed to address this issue. Instead of simply applying quantization to the pre-trained weight like QLoRA, LoftQ utilizes both quantization and low-rank approximation and accounts for the subsequent LoRA fine-tuning process which leads to notable performance increase and results in negligible computational cost.

Adapters have significant implications for small businesses that do not have the massive financial backing required to train a Frontier model with the cost of training GPT-4 cited as around \$63mil (Schreiner [2023]) and costs only expected to rise from there. Thus the appeal of PEFT is the ability to potentially tailor a model to a specific task with a specific dataset and achieve performance rivaling a frontier model in a much less computationally intensive way

(Predibase). While a fine tune does not require the huge corpus of data or the same amount of training time as the pre-training phase it does still require enough GPU memory to fit all the parameters of the model, their gradients, the tokenized data, optimizer states and activation's. In the case of something like Mistral 7b this is over 120gb of memory. At the current price of GPUs this would require significant investment. As a counter point the training platform Predibase trained 25 adapters on Mistral 7b and was able to achieve frontier level performance at the cost of \$8 per model (see: Predibase). Thus LoRA addresses a major limit of what had been the the current practice, full fine tuning, by significantly reducing the amount of compute to achieve a similar result.

1.3. Training Data

Selecting a dataset proved to be the first major challenge of this project. We were unable to find a labeled dataset that did not also include common benchmark questions. This limitation drove the the decision to hand craft a suite of benchmark questions which allowed us to use publicly available data. We select what is known as the "Finance-Alpaca" dataset (Bharti) which is a combination of the popular Alpaca instruction tuning dataset from Stanford (85k downloads/mo on Hugging Face, Taori et al. [2023]) and the FiQA benchmarking questions which is a popular suite of questions for financial benchmarking (Lab and Technology). Our dataset consists of roughly 69k rows of instruction and output pairs, approximately 52k from Alpaca, 16k from FiQA and 1.3k that were also generate from GPT-3.5. Some questions provided additional context in the form of an input which is a document the question is asking to summarize or extract information from. In those situations the input was merged with the question and our base prompt template, this was done to simplify the training architecture so that only input and output are required. See appendix A for samples.

During training all data was tokenized with the Mistral 7b tokenizer. Table 1 shows the tokenized length of instruction output pairs in the form of percentiles. We examined filtering out data with tokenized lengths over the 99th percentile due to the long tail of the distribution. However we encountered issues with trying to load a custom dataset using the HuggingFace PEFT and Transformers infrastructure and instead opted to truncate on the fly during training to 550 tokens. We reasoned that this should not eliminate any data from the instruction and that the majority of the knowledge during the response should be captured towards the beginning of the response.

Benchmark and training prompts were all given with the base template of "You are a friendly financial advisor. Your job is to help people answer financial and related question. Make sure when you answer questions you show the steps

	instruction	output
average	17	106
min	2	0
max	628	3688
median	15	68
75th percentile	20	128
90th percentile	29	249
95th percentile	37	362
99th percentile	71	694

Table 1. Summary statistics of training data after tokenizing

you took to arrive at your answer. Do your best to reason step by step. If you don't know the answer, say that you don't know.". The benchmark consisted of 9 handcrafted questions, 3 each in math, theory and summarizing.

2. Approach

2.1. LoRA Background

The core concept of LoRA is to take advantage of the properties of low rank matrices where rows or columns within the matrices are not linearly independent and therefore translate to redundant features or information that could be decomposed. LoRA assumes that the base model's weight matrices are full rank with low intrinsic dimension. The authors of the the LoRA paper also suggest that the update to the weights have a low intrinsic rank. This allows for a decomposition into a smaller A and B matrix which then form the adapter. The new forward pass then becomes EQ 1, where A is initialized with Gaussian noise and B is initialized as zeros. The overall result is that the trainable parameters are in the range of 0.1-1% of the original parameter count. The practitioner selects the amount of trainable parameters by choosing the rank. The α term is introduced which scale the scales the gradient update. Alpha is similar to a learning rate and while the authors recommend setting it equal to rank there are many opinions on what a suitable value is in practice.

$$h = W_0x + \Delta Wx = W_0x + BAx \quad (1)$$

While QLoRA was cover in the earlier section; a treatment of LoftQ (LoRA-Fine-Tuning-aware Quantization) is still required. The authors of the LoftQ paper provide a very good description in our opinion. "It is designed specifically for pre-trained models that require quantization and LoRA fine-tuning. This framework actively integrates low-rank approximation, working in tandem with quantization to jointly approximate the original high-precision pre-trained weights. This synergy significantly enhances alignment with the original pre-trained weights as illustrated in Figure 2. Consequently, our method provides an advantageous initialization point for subsequent LoRA fine-tuning,

leading to improvements in downstream tasks". See [Li et al. \[2023\]](#) for further reading on the subject.

2.2. Experiment Goals and Setup

Our experimental setup was designed with two sub goals in mind on top of the larger meta goal. Sub goal 1 was to understand if changing the LoRA rank had any effect on the model performance when ran through our benchmarking suite. Sub goal 2 was to examine if the different LoRA methods had any influence on performance at inference time. As such we trained 6 different models using the HuggingFace transformers library on the PACE cluster. All models were trained on a single H100 GPU. We believe that direct comparison of adapter frameworks is a novel concept and is still a field of open research.

As discussed in the data section data is preprocessed into a 2 column dataframe and then tokenized on the fly during training. Code was inspired by ShawhinT's youtube videos on QLoRA training and the repository can be found here [ShawhinT](#). The code was modified to take in the finance dataset used for the training and split it into a training and testing set. Modifications were also made to use the base model instead of the instruct version of the model. The HuggingFace BitsAndBytes library was used to quantize the model before performing LoRA in the QLoRA training. For the LoRA adapters, the code was modified to remove any quantization and use the base model. For LoftQ, the HuggingFace documentation on LoftQ was used to modify the code to first perform quantization and bring in LoftQ configurations, after which regular LoRA was performed. The code was also modified to ensure that the correct versions of the model are being saved. This was then executed via a batch script on the PACE cluster. Expectations based on the literature and marketing around LoRA and H100 GPUs was that training should be very fast, on the order of 2-3 hours.

The loss function used is the cross-entropy loss. It is calculated based upon the discrepancy between the predicted probability of words in the sequence and the actual distribution from the dataset. This loss is then minimized during training using the Adam optimizer. The weights of the base Mistral 7b were frozen and only the adapter module weights were tuned using this optimizer.

Table 2 below shows the final design of experiments. The expectation was that by varying the LoRA type, rank and alpha we should see different results in terms of training time, loss, memory usage and evaluation performance.

2.3. Risks and Potential Issues

Anticipated problems were data bleed (test data mixing in with pre-training data), lack of compute resources and having to do too large of a design of experiments to see promising results. Data bleed was mitigated by creating

Model	Rank	Alpha	Quantizing	Params
LoRA	8	16	bf16	3,407,872
LoRA	16	32	bf16	6,815,744
QLoRA	8	16	fp8	3,407,872
QLoRA	16	32	fp8	6,815,744
LoftQ	8	16	fp8	3,407,872
LoftQ	16	32	fp8	6,815,744

Table 2. The design of experiments the team used to try and train a finance adapter

original questions for the LLMs that are at a minimum syntactically different from any question found in common validation sets such as FiQA. See appendix E for the full question list. While LoRA models are faster and less compute intensive to train compared to a full model, a GPU is still needed. The group pursued 3 options in parallel, PACE-ICE, Colab and Predibase. After comparing level of effort and flexibility of all 3 platforms the team decided to use to use PACE-ICE for training the models.

All models were expected to struggle with the math problems as has been documented in numerous places (Srivatsa and Kochmar [2024]), however the expectation was that the LoRA process would improve capabilities here as finance and math typically go hand in hand. Most labeled data sets (including ours) are scraped from Reddit. Typically this is done by turning the original question into the query for the model and then choosing the most upvoted post for the response. However Reddit is not a scholarly source and often contains factually incorrect or over opinionated information.

We leave a full discussion of the issues encountered to the experiment section. In summary, we encountered infrastructure issues, unexpectedly long training time, documentation problems, and great difficulty separating influencer quality material from reliable and useful sources.

3. Experiments and Results

3.1. Evaluation Methods

Testing is done in a zero-shot fashion with a consistent prompt designed to evoke chain of thought (COT) style reasoning (Wei et al. [2023]), see appendix A. COT has been shown to improve model capability on tasks requiring multiple steps to solve which is common for math related problems. Claude 3 Opus being a frontier model was expected to have the best performance only being rivaled by the AdaptLLM Finance Chat model which has been specifically tailored to the typed of questions we are asking. Success was defined as the adapter models matching the Frontier models when the scores from each section are tallied.

As part of hyper parameter tuning process we wanted to see if alpha had any significant impact on the quality of the responses since our primary focus was rank. We trained 4x

rank 8 QLoRA models on Predibase sweeping alpha across 1, 2, 4, and 8. This was the point we found that Predibase would not allow us to use our adapters on top of Mistral 7b base. We found during benchmarking that all models scored the same and gave similar quality of answer. However we cannot attribute this to any aspect of the training setup with certainty due to the very strong bias that an instruction tuned model has. For remaining experiments we used alpha 2x rank. See Appendix D for data from this experiment.

3.2. Quantitative Results

The experiments performed show the efficiency of training adapters over a full fine-tuning of the model, both in space used by the adapters and time taken to train. We attempted to do a full fine tune of Mistral 7b however we found this takes 31 hours. The adapters take much less time to tune as shown in Table 3. On the Math section of the finance questions, all the models performed poorly. The Theory benchmarks showed that the adapter modules can compete with the the base models with adapter performing as well as the AdaptLLM Finance Chat (Cheng et al. [2024]) model in theory. In the Summary benchmarks, the adapter models once again could compete with the reference models.

Reduced training time shown in Table 3 is in line with expectations due to the networks having significantly fewer parameters though not to the degree expected. Claude 3 Opus, for example, is a state-of-the-art model with 2 trillion parameters. Our adapters were not only smaller, but were also trained on the much smaller Mistral 7b. Therefore, the dominance of Claude was seen as expected. Our adapter all exhibited poorer quantitative score compared to a baseline instruction tuned model and to frontier models. We attribute this to the same reasons the qualitative results were poorer which is discussed below.

Loss during training is shown in Table 9. We see that loss only changed significantly in the last model in the table due to a learning rate that was set too high. In all other examples the very low loss scores and small changes indicate that the models very closely fit the data. This theory is supported by how closely the responses from the models match the responses from the training data.

3.3. Qualitative Observations

The qualitative results of the experiments are also studied. On the Math benchmarks, all the models failed. Though the model could perform math operations, they were not accurate either using numbers that had no relation to the problem statement or were correct but then had a result that did not agree with the operation performed (ex: $2+2=5$). This was not surprising but disappointing given the expectations that had been set with regards to adapter.

The adapter models exhibited repetitive speech that is

Model	Model Type	Training Method	Params	Train Time	Model Size
Claude 3 Opus	Proprietary	Pre-trained/RLHF	Estimated 2 tril	—	—
Mistral 7b Instruct V2	Open Weight	Pre-trained/RLHF	7bil	—	15gb
AdaptLLM Finance Chat	Open Weight	Continued pre-training	7bil	—	25gb
Predibase QLoRA 256	Adapter	Supervised SGD	109,051,904	8	1.1gb
QLoRA Rank 8	Adapter	Supervised SGD	3,407,872	11	12mb
QLoRA Rank 16	Adapter	Supervised SGD	6,815,744	11	24mb
LoftQ Rank 8	Adapter	Supervised SGD	3,407,872	7	12mb
LoftQ Rank 16	Adapter	Supervised SGD	6,815,744	7	24mb
LoRA Rank 8	Adapter	Supervised SGD	3,407,872	7	12mb
LoRA Rank 16	Adapter	Supervised SGD	6,815,744	7	24mb

Table 3. Overview of model types, parameter counts and training times. Training times not included for models not trained by the team. We see that LoRA creates models significantly smaller in size than the base models

Model R, α	Math	Theory	Summary	Total
LoRA (8,16)	0	2	2	44%
LoRA (16,32)	0	2	2	44%
QLoRA (8,16)	0	2	2	44%
QLoRA (16,32)	0	2	2	44%
LoftQ (8,16)	0	2	3	44%
LoftQ (16,32)	0	3	1	44%
Predibase (best)	0	3	2	55%
AdaptLLM	0	2	3	55%
Mistral 7b inst	0	3	3	66%
Claude 3 Opus	1	3	3	77%

Table 4. Question Benchmark Results. Claude was the only model that showed any resonable performance on the math questions

Model	Start Loss	End Loss	Iteration/Sec
LoRA (8,16)	1.136	1.048	10.11
LoRA (16,32)	1.169	0.987	10.167
QLoRA (8,16)	1.178	1.111	6.634
QLoRA (16,32)	1.182	1.079	6.619
LoftQ (8,16)	1.177	1.105	10.142
LoftQ (16,32)	1.178	1.079	10.218
QLoRA (16, 32)	14.726	239mil	6.632

Table 5. Start and end evaluation loss for each model after 5 epochs. All models use a learnign rate of 0.0001 except for the model in bold which use an lr=0.0075

reminiscent of Reddit style posts. We attribute this to over fitting and low quality source data. Selected prompts and answers are provided in appendix B. We suggest the poor data quality hypothesis on the basis that QA pairs are just scraped from the internet and not validated for factuality. We suggest the overfitting hypothesis as the loss function is trying to match the target word for word. Since the base model has not been instruction tuned this means that it has now been biased significantly towards generating text that precisely matches the responses word for word. A way around this would be to obfuscate the question answer pairs. For example a response to "why is the sky blue?" could be "due to scattering by the atmosphere" or just as easily "light

is refracted by the particle in the atmosphere causing a color shift". This would be a form of data augmentation.

A key takeaway from this work is that the adapter modules have difficulty going beyond the capabilities of a regular LLM. From the Math section of the benchmarks, it can be seen that the adapter trained models performed poorly, just like the fully fine-tuned models. While we did not succeed in training an adapter to frontier level we were able to show that for small benchmarks LoRA training method, rank, and alpha are small factors in the quality of the result. Our experiment suggests that training data quality and the choice of base model play the greatest role in the effectiveness of an adapter.

3.4. Issues Encountered

As referenced in other sections training time we unexpectedly long significantly limiting the amount of experiments we could successfully run. Our initial plan started off with us using Predibase for our experiments only to find that it will not let you run inference on Mistral 7b base. Further Predibase did a major refactor on it's API mid project resulting in the need to rewrite code. Finally they abstract away many of the hyperparameters making it difficult to run apples to apple experiments. Ultimately Predibase was relegated to just doing an alpha sweep experiment to see if we could find any noticeable changes from that hyperparameter (we didn't). Finally they also removed learning curves and loss reporting.

The team encountered several issues with CUDA and the HuggingFace library when trying to use the PACE cluster. For example only one of our team members was able to successfully configure a python environment and submit batch jobs. This is in spite of us trying to replicate environments, sbatch scripts and python code. One team member consistently experience CUDA out of memory errors in spite of using tricks like gradient checkpointing that should have freed up enough space given we were using H100s. Another team member attempted to use the BitAndBytes library and consistently encountered C library errors. All team mem-

bers encountered issues with the Accelerate library preventing us from effectively parallelizing.

4. Conclusion

In this paper we showed that LoRA adapter can significantly lower the time to train by leveraging existing models. We also showed that very careful treatment of the data and the hyper parameters is needed to achieve frontier level performance. Popular discussion and marketing would suggest that training and adapter for that sort of performance is quite trivial, we find this to be untrue.

We also show that choice of quantizing scheme does not impact question answer performance at our level. Similar to most Machine Learning model development, many architectural configuration or parameterization are highly experimental. With quantized training being a relatively new framework the team feels that the landscape needs more time to settle and a winner to be declared before these methods see mass appeal. There are simply too many different options with their own unique pitfalls to know which to select and how to use it properly unless you are a very active practitioner in this particular field. The failure of our adapters to improve performance and the broad landscape provide ample space for future work.

References

- Gaurang Bharti. Gbharti/finance-alpaca · datasets at hugging face. URL <https://huggingface.co/datasets/gbharti/finance-alpaca?row=3>. 2
- Daixuan Cheng, Shaohan Huang, and Furu Wei. Adapting large language models via reading comprehension, 2024. 4
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023. 1
- Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morroni, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp, 2019. 1
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. 1, 2
- Social Lab and Language Technology. Salt-nlp/flue-fiq · datasets at hugging face. URL <https://huggingface.co/datasets/SALT-NLP/FLUE-FIQA>. 2
- Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, and Tuo Zhao. Loftq: Lora-fine-tuning-aware quantization for large language models, 2023. 2, 3
- Predibase. Lora: Fine-tuned open source llms that outperform gpt-4. <https://predibase.com/blog/lora-land-fine-tuned-open-source-llms-that-outper>. Accessed: 03-22-24. 2
- Maximilian Schreiner. Gpt-4 architecture, datasets, costs and more leaked, Jul 2023. URL <https://the-decoder.com/gpt-4-architecture-datasets-costs-and-more-leaked>. 2
- ShawhinT. Youtube-blog/llms/qlora at main · shawhint/youtube-blog. URL <https://github.com/ShawhinT/YouTube-Blog/tree/main/LLMs/qlora>. 3
- KV Aditya Srivatsa and Ekaterina Kochmar. What makes math word problems challenging for llms?, 2024. 4
- Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023. 2
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. 4

5. Work Division

Charles wrote 55% of the paper, created all tables, 90% of citations, and found and added all figures and appendices. Yohann wrote an additional 25% of the paper and was very focused on executing training runs when other team members started encountering issues with their setup. Winston wrote the remaining 10% and contributed the work related to LoftQ. At the start of the project all team members contributed to the proposal and See Table 6 for more info.

Student Name	Contributed Aspects	Details
Charles Kniffin	Data Selection, LoRA training, Predibase Training Benchmarking Creation and Execution, Paper Writing	Performed project management work and ran team meetings.
Winston Lai	LoftQ research, Paper Writing, Colab testing	Provided a voice of reason and kept Charles from going off the deep end
Yohann	PACE Cluster Expert, QLoRA Training, Base Code Development, Paper Writing	3rd shift and awesome doer of tasks

Table 6. Contributions of team members.

6. Appendices

6.1. Appendix A: Base Prompt and Selected Question Answer Pairs

Table 8 provides several examples of question answer pairs. Notice that wording, answer length and quality significantly vary. Wording and length can be thought of as a form of regularization however quality must be maintained for the model to provide accurate outputs. The base prompt was of the form:

”You are a friendly financial advisor. Your job is to help people answer financial and related question. Make sure when you answer questions you show the steps you took to arrive at your answer. Do your best to reason step by step. If you don’t know the answer, say that you don’t know.

Instruction: How does a business compute net income?

Answer:

Input	Output
You are a financial analyst designed to answer questions about business and finance. Your job is to reply to questions about finance topics and provide advice. Instruction: How does a business compute net income?	You've got most things right, I just needed to make few edits: Cash Flow from Operations is Net Income plus Difference between the Operating Assets and Operating Liabilities. Cash Flow from Operations is similar but not equal to the Net Income. You missed Operating Expenses and it was the reason the numbers on your answer were wrong. To understand why Operating Expenses need to be taken out from Net Income, look at this simple example: you have operating expenses like rent, salaries, etc. and net income in a given period. You sold a car for 8000, you received 5000 from rent and you paid 3000 for salaries. You still have operating cash in your company. However, if you then used all this cash to buy a new car for 6000, you will still have the net income of 8000 but your operating cash is 0. This happens because this money went into the purchase of assets, not in cash. The net income is only one aspect of a company's cash position. This is why Operating Expenses need to be taken out. Operating Income is equal to Cash Flow from Operations if everything else that you did in the period did not affect this amount. However, in reality it is not always the case. This is why Cash Flow from Operations is also calculated. For a better understanding, read on wikipedia. When I calculate my operating expenses, should I include the interest on my assets? No. The only interest included in the Operating Expenses are interest paid. Interest received is the only item that should be included in an investment account as Investments received. Edit: After the discussion in the comments, I would like to add one more thing to clarify why Operating Expenses needs to be taken out, a concrete example. If you look at the Cash Flow statement, you will see that all the differences between the change of the Operating Assets and Operating Liabilities will be included there, including the Interest Expenses and Receipts. Therefore, if you subtract the Operating Expenses from the Operating Income (which is the Total Comprehensive Income, on your example this is 3000), the amount you get is exactly the same as the Net Cash provided by Operations on the Cash Flow statement. However,
If I take out a \$35000 loan with a 5% APR how many \$500 payments would it take to pay it back?	82 or 83 (due to partial payment for the last one)
Assuming an initial investment of \$100,000, an expected cash flow of \$7000 per month, an interest rate of 7% and 3 years before the project is complete what is the net present value?	-31430
You want to save up over 4 years for you child's college. If the cost of attendance is \$100000 and you deposit money each month into a savings account with a 4% annual interest rate, how much do you have to deposit each month?	1926.12
What is the rule of 72?	The Rule of 72 is a quick, useful formula that is popularly used to estimate the number of years required to double the invested money at a given annual rate of return.
What is CPI and how does it tell us what the inflation rate is?	The Consumer Price Index (CPI) is a measure of the average change over time in the prices paid by urban consumers for a market basket of consumer goods and services.

Table 7. Sample prompt with response from testing of Lora Rank 8. Additional responses omitted for space reasons

6.2. Appendix B: Selected Prompts and Responses

Table 7 shows the type of prompt the model is given during testing and a typical response. Notice how the long rambling answer resembles the training data. The model has decided not to produce an EOS token and runs until it reaches its generation limit.

Input	Output
Why does it matter if a Central Bank has a negative rather than 0% interest rate?	That is kind of the point, one of the hopes is that it incentivizes banks to stop storing money and start injecting it into the economy themselves. Compared to the European Central Bank investing directly into the economy the way the US central bank has been doing. (The Federal Reserve buying mortgage backed securities) On a country level, individual European countries have tried this before in recent times with no noticeable effect.
Approximation of equity value for company in default	Generally "default" means that the company cannot pay off their debts, and since debt holders get paid before equity holders, their equity would be effectively worthless. That said, companies can emerge from Chapter 11 bankruptcy (reorganization) and retain equity value, but it is rare. Most times, stocks are de-listed or frozen on stock exchanges, and company's reorganization plan will cancel all existing equity shares, instead focusing all of their attention on paying back as much debt as possible. If the company issues new equity after reorganizing, it might provide a way for holders of the original equity to exchange their shares for the new equity, but it is rare, and the value is usually significantly less than the value of the original equity.
Is it true 90% of investors lose their money?	The game is not zero sum. When a friend and I chop down a tree, and build a house from it, the house has value, far greater than the value of a standing tree. Our labor has turned into something of value. In theory, a company starts from an idea, and offers either a good or service to create value. There are scams that make it seem like a Vegas casino. There are times a stock will trade for well above what it should. When I buy the S&P index at a fair price for 1000 (through an etf or fund) and years later it's 1400, the gain isn't out of someone else's pocket, else the amount of wealth in the world would be fixed and that's not the case. Over time, investors lag the market return for multiple reasons, trading costs, bad timing, etc. Statements such as "90% lose money" are hyperbole meant to separate you from your money. A self fulfilling prophesy.... (continues)

Table 8. Examples of the prompt/answer pairs

6.3. Appendix C: Start and End Losses for All Models

Table 9 shows the start and end training loss for each model. We attribute the lower iterations/second for QLoRA to a quirk of implementation in the HuggingFace Library.

Model	Start Loss	End Loss	Iteration/Sec
LoRA (8,16)	1.136	1.048	10.11
LoRA (16,32)	1.169	0.987	10.167
QLoRA (8,16)	1.178	1.111	6.634
QLoRA (16,32)	1.182	1.079	6.619
LoftQ (8,16)	1.177	1.105	10.142
LoftQ (16,32)	1.178	1.079	10.218

Table 9. Start and end evaluation loss for each model after 5 epochs

6.4. Appendix D: Alpha Sweep Results Summary

A QLoRA alpha sweep was done on a rank=8 adapter to see if there are any impacts.

α	Math	Theory	Summary	Total
1	0	3	3	66%
2	0	3	3	66%
4	0	3	3	66%
8	0	3	3	66%

Table 10. Sweeping alpha had no major impact