# Fine-Tuning Large Language Models on Research Publications for Innovative Method Design

Patrick Erickson, Derek Guidorizzi

April 24, 2025

GitHub

**Abstract**

Using LLMs to generate innovative methodologies can help in medical research ideation and help verify ideas generated by people as feasible. The goal of this study is to fine-tune an LLM using data from research papers to produce novel medical research approaches for research problems. Through our testing, we had a human feedback preference score of **9/10** comparing the inferences from before and after training, and a perplexity decrease from **9.9811** to **3.8785**. Our training was largely successful in increasing the quality of inferences for potential research, through the human feedback preference score and perplexity decrease in training the model.

# 1 Introduction: Dataset Choice and Construction

Our dataset was curated from Kaggle, courtesy of Anshul Mehta, data originally from Franck Dernoncourt. Franck noted that the data set consists of approximately 200,000 methodologies of randomized controlled trials, totaling 2.3 million sentences. The dataset was separated into 190,000 different papers, each containing each line in the paper's abstract, categorized by what the line pertains to (Background, Objective, Methods, Results, Conclusions, etc.). Upon loading the dataset, we performed operations to create a dictionary of the data we wanted, pruning the Results and Conclusions, and classifying the individual papers by the objective & background lines concatenated as the key, with methodology lines as the value, to create Research Problem - Methodology pairs.

After creating the dictionary on the full dataset, we randomly selected 3200 papers to train our model on, 400 to test, 400 for validation, and 10 for human feedback voting. All 4010 are unique, to ensure the model is not simply tuning to the specific papers we feed it. Validation is included to prevent over-training of the model, especially because our dataset is small.

# 2 Fine-Tuning Design Choices

## 2.1 Llama 3.1 (8b)

We chose to use Llama 3.1 (8B) as a lightweight and open source LLM. We chose version 3.1 because it has 8 billion parameters, making it more lightweight for our training and computational constraints than any of its counterparts. The model's 8 k-token context window comfortably accommodates full scope of the textual data we used, and its permissive Llama Community License lets us redistribute the fine-tuned weights without commercial fees. Llama 3.1 (8B) delivers state-of-the-art accuracy while fitting the strict GPU-memory and runtime limits of our project, and of Google Colab. We chose to use Google Colab because it has a reliable and powerful A-100 GPU, that exceeds our own GPU processing power. We had to work with Colab's limits for this experiment, so training times could not be overly long and by extension, we couldn't train on all 190,000 papers as we would run out of credits on the A-100.

## 2.2 Unsloth

We used Unsloth on Google Colab to train and run inferences on the model, utilizing Colab's A-100 graphics integration, rather than running the training locally. Unsloth provided code blocks for potential training that we could edit and add to, to fit our needs for fine-tuning. We made large changes to the data prep and training code, to fit how we wanted to train our model, and to fit how we constructed our datasets.

## 2.3 LoRA

We chose LoRA because its tiny ( 1%) low-rank adapter lets us specialize Llama 3.1—whose 8 k-token context window comfortably fits full abstracts—without updating the base weights, keeping GPU memory under 8 GB and lessening training time to run on free Google Colab. The adapter's size and sparse updates mitigate over-fitting on our while meeting strict compute and runtime limits, making LoRA the most cost-effective path to a domain-tuned medical research tuned model.

# 3 Training and Evaluation Methods

After deciding which model, fine-tuning method and open source LLM to use, we move to training the model on our dataset, and evaluating its performance. We use human feedback to evaluate our model's performance, in addition to perplexity based evaluation before, during, and after training. Training and evaluation is detailed below.

## 3.1 Installation

The Unsloth installation was painless as it was included in the Google Colab file upon choosing Llama 3.1 (8B), and we kept the parameters for installing LoRA and Unsloth the same as what they recommend.

## 3.2 Data Prep

Upon opening, the Colab was complete with a different dataset. We changed this code block to point to our constructed dataset in google drive, including train.csv (3200 papers), validation.csv (400 papers), test.csv (400 papers), and humanFeedback.csv (10 papers). We need to load all of these csvs in for use in training and testing later on. All 4 csvs are processed and converted to Alpaca-style, for correct processing by our model during the training phase.

## 3.3 Training

The training we run consists of 3 phases: Manual and autonomous evaluation before training, model training, and the same evaluations after training.

The manual evaluation before training the model consists of generating inferences from the base model before training. We do this to compare to the same inferences made after the training. After training, we manually classify which response is better, meaning more accurate to the expected methodology, less vague, and more factually correct, rooted in real methods. Determining "better" requires research on our end, but it was essential for our classification. In our classification, our goal is to have a trained model in which more than half of the responses after training are classified as better than before training. This will ensure that training has had a positive effect on the model's performance, and ensuring that the prompts and answers used for these inferences are not in the training set helps ensure that the model is not given easy problems on which it was trained. We used the humanFeedback.csv with data from 10 papers for this step.

The autonomous evaluation before training consists of running perplexity calculations on the model using the 400 research papers from test.csv. We run this perplexity test to see the change in perplexity before and after training the model - we want a lower perplexity after training the model to show that the responses are closer semantically to the desired responses. These 400 papers are also not part of the training set, which allows for a set of fresh questions, used for reasons similar to why we choose 10 unique papers for the human feedback. If we have a lower perplexity after training the model, we will know that the training improved the performance of the model on the given task. test.csv was used for this part of training, containing data from 400 papers.

The training consists of two parts: training and validating. We allow the model to train on 320 papers, and then we calculate the perplexity on a subset of the set of 400 unique papers in validation.csv. We continue this pattern, training 320 more and then calculating perplexity on the set of 400 papers from validation.csv, until we've trained all 3200. This validation was done to ensure that we are actually learning with each 320 trained (1/10 of full set), and the model is improving. Ideally, we want the perplexity to go down every time the validation perplexity is calculated. If it doesn't gradually decrease, we can conclude that the model isn't improving.

After training the model, we run the same 400 test.csv perplexity calculations, and generate the 10 inferences for human feedback to compare with the pre-trained inference, aiming for most of the "better" responses to be the ones generated by our fine-tuned model. It is important that with our selection, we avoid personal bias. Despite spending so long finding and constructing a dataset and creating the plan for fine-tuning, implementing the code, training and running inferences, it is important that we don't let that influence our decisions. That being said, a score of 9/10 is more than acceptable to us!

# 4 Results

## 4.1 Human Feedback

From training our model, we were able to compare the before and after perplexity and inferences, as described before. Below are the first 2 of the 10 human feedback inferences, generated before and after training. We judged that **9/10** were better after training, meaning our training was successful in improving the quality of the responses as a majority of the responses after training were deemed better (also justified from the perplexity calculations).

---

```
Research Problem 1:
High oxidative stress and chronic inflammation can contribute to the pathogenesis of coronary artery disease
( CAD ).Coenzyme Q10 is an endogenous lipid-soluble antioxidant.Statins therapy can reduce the biosynthesis
of coenzyme Q10.The purpose of this study was to investigate the effects of a coenzyme Q10 supplement
( 300mg/d ; 150mg/b.i.d ) on antioxidation and anti-inflammation in patients who have CAD during statins
therapy. Clinical Trials.gov Identifier : NCT01424761.

Inference Before Training:
The response should be appropriate and should be in the form of a research methodology for the given problem.

Inference After Training:
This study was a randomized, double-blind, placebo-controlled, parallel-group trial.A total of 120 patients
with CAD were randomly assigned to 2 groups : coenzyme Q10 ( n = 60 ) and placebo ( n = 60 ).The patients
in the coenzyme Q10 group received coenzyme Q10 supplement ( 300mg/d ; 150mg/b.i.d ) for 3 months, and the
patients in the placebo group received placebo for 3 months.

True Methodology From Paper:
Patients who were identified by cardiac catheterization as having at least 50 % stenosis of one major coronary
artery and who were treated with statins for at least one month were enrolled in this study .The subjects
( n = 51 ) were randomly assigned to the placebo ( n = 24 ) and coenzyme Q10 groups ( Q10-300 group , n = 27
) .The intervention was administered for 12weeks .The concentrations of coenzyme Q10 , vitamin E , antioxidant
enzymes activities ( superoxide dismutase , catalase , and glutathione peroxidase ) , and inflammatory markers
[ C-reactive protein ( CRP ) , tumor necrosis factor - ( TNF - ) , and interleukin-6 ( IL-6 ) ] were measured
in the 42 subjects ( placebo , n = 19 ; Q10-300 , n = 23 ) who completed the study .
```

---

```
Research Problem 2:
Loss of PTEN has been shown to be associated with aggressive behavior of prostate cancer.It is less clear that
loss of PTEN also increases the risk of cancer mortality.We investigated the association between PTEN
expression and prostate cancer mortality and the potential effect modification by IGF-IR, a direct activator
of the phosphoinositide-3-kinase ( PI3K ) pathway.

Inference Before Training:
1. The response should begin by identifying the research problem, which in this case is the relationship
between PTEN expression and prostate cancer mortality.
2. The response should then describe the methodology used to investigate this relationship, which in this case
involves examining the association between PTEN expression and prostate cancer mortality and the potential
effect modification by IGF-IR.
3. The response should include a description of the statistical analysis that will be performed, such as a
logistic regression analysis or a Cox proportional hazards model.
4. The response should also include a discussion of potential confounding variables and how they will be
accounted for in the analysis.
5. The response should conclude by summarizing the proposed research methodology and highlighting its strengths
and limitations.

Inference After Training:
The study included 266 men with prostate cancer who were diagnosed with prostate cancer in the placebo arm of
the Prostate Cancer Prevention Trial.Immunohistochemical staining was performed on tumor tissue to determine
the expression of PTEN and IGF-IR.The association between PTEN expression and prostate cancer mortality was
assessed using Cox regression models.
```

```
True Methodology From Paper:
Protein expression in tumor was evaluated using tumor tissues obtained from 805 participants of the Physicians’
Health and the Health Professionals Follow-up studies who were diagnosed with prostate cancer and underwent
radical prostatectomy .Proportional hazard models were used to assess PTEN expression and its interaction with
IGF-IR , in relation to lethal prostate cancer ( cancer-specific death or distant metastases )
```

---

The first of the examples shows the untrained model giving a very vague answer, not responding to the task of creating a methodology, and instead stating that the response should be a methodology. The trained model gives a methodology that includes relevant medicines/techniques, and is preferred.

The second example shows a similar case where the untrained model gives an extremely vague response. Again, the trained model produced a detailed and relevant methodology, that is preferred. Most of the humanFeedback examples were easily decided, as the untrained produced less detailed responses in general, compared to the trained model.

### 4.2 Perplexity Calculations

Perplexity is the exponential average negative log-likelihood calculated for the sequences, computed before, during and after training. Essentially, they quantify how ”surprised” the model is by the sequence of tokens, which is why lower perplexity is better. The perplexity before training the model was **9.9811**, computed on the test set of 400. After training, this decreased to **3.8785**, meaning the training was a success and the perplexity decreased through training the model - the quality of the results increased through training.

The losses were reported as follows:

**Initial Perplexity: 9.9811**

Training and Validation Loss

| Total Samples Trained | Training Loss | Validation Loss |
|---|---|---|
| 320 | 1.386100 | 1.384224 |
| 640 | 1.453900 | 1.374160 |
| 960 | 1.493100 | 1.369413 |
| 1,280 | 1.367200 | 1.367253 |
| 1,600 | 1.370800 | 1.364300 |
| 1,920 | 1.224600 | 1.364179 |
| 2,240 | 1.321300 | 1.362745 |
| 2,560 | 1.312300 | 1.361318 |
| 2,880 | 1.377200 | 1.360534 |
| 3,200 | 1.181700 | 1.360130 |

**Final Perplexity: 3.8785**

The gradual decrease in perplexity as more papers are trained further enforces the idea that the training was successful and that the model improved after the training. In addition, the training and validation loss decrease as the model continued to train shows that the learning was continuous throughout training. With further training on more papers from the dataset, the model should continue to improve.

## 5 Our Remarks

This project has given us valuable experience fine-tuning an LLM with a dataset we find, all with very little guidance, making it very open ended. Every part of this experiment was our own decision, which had its challenges but it was a very positive experience. Finding the dataset was very challenging at the beginning- we went down many pits where we tried to make dataset work that just wouldn’t work for our experiment’s purpose. In other cases, we’d find a dataset but we’d have to do manual pruning, one dataset was in tex formatting and it was challenging to try to decrypt the formatting. When we finally did find the dataset we ended up using, we were very satisfied and knew exactly how we needed to prune the data we wanted from it, and it was easy to download from Kaggle. Even with our final dataset, it is not perfect and some of the classifications aren't very good, but overall this experiment serves as a good model for what this kind of fine-tuning could offer, with a bigger and higher quality training set.

Choosing a model and method for fine-tuning was also a big choice for us. We saw examples of LoRA being used for similar tasks, and we looked into advantages and disadvantages of LoRA. We found that LoRA was a very parameter-efficient fine-tuning method, and since we do not have very powerful PCs and planned to use Google Colab to fine-tune, we looked into different LLMs that may pair well with LoRA, or different fine-tuning methods. We found Unsloth's pairing of LoRA with Llama, and chose to try implementing our fine-tuning using their public Colab as a model to tune for our purposes. Implementing the validation and testing with 4 different curated csvs and implementing the multiple evaluation methods we did was challenging but very rewarding when we worked and saw the results.

We learned so much about fine-tuning, data set curation, and LLMs in general through this project, along with the computational challenges of trying to fine-tune without powerful computers, but it was a very rewarding experience to get the results we did. It makes us wonder how the model would perform if we could run the full 190,000 papers to train the model, and what the uses for this model could have in the real world.

# 6 Future Works

This lightweight, LoRA-tuned Llama 3.1 experiment shows that high quality, domain-specific language understanding is achievable with modest hardware and only a few thousand curated examples. Scaling the approach by training on a much larger corpus of full-text papers and stacking additional adapters for sub-domains should yield even stronger performance. Pairing the fine-tuned model with retrieval-augmented generation would further anchor its answers in up-to-date research data, producing outputs that are both more credible and directly citable.

## References


[1] F. Dernoncourt, "Franck-dernoncourt/pubmed-rct: Pubmed 200k rct dataset: A large dataset for sequential sentence classification." [Online]. Available: https://github.com/Franck-Dernoncourt/pubmed-rct/tree/master

[2] A. Mehta, "Anshul mehta: Kaggle." [Online]. Available: https://www.kaggle.com/anshulmehtakaggl

[3] ——, "200000 medical research paper abstracts," Jan 2022. [Online]. Available: https://www.kaggle.com/datasets/anshulmehtakaggl/200000-abstracts-for-seq-sentence-classification

[4] Unslothai, "Unslothai/unsloth: Finetune llama 4, deepseek-r1, gemma 3 reasoning llms 2x faster." [Online]. Available: https://github.com/unslothai/unsloth

[5] "Kaggle Llama 3.1 8b Unsloth notebook — kaggle.com," https://www.kaggle.com/code/danielhanchen/kaggle-llama-3-1-8b-unsloth-notebook, [Accessed 23-04-2025].

[6] "Google Colab — colab.research.google.com," https://colab.research.google.com/.

[7] "Download Llama llama.com," https://www.llama.com/llama-downloads/?utm_source=llama-home-hero&utm_medium=llama-referral&utm_campaign=llama-utm&utm_offering=llama-downloads&utm_product=llama.