EXPERIMENTATION WITH LLM FINE-TUNING

Takeaways from completed project:

🚀 For crypto trading applications, fine-tuned open-source LLMs via PEFT on context-rich, labelled data in Google Colab.

🔧 While the fine-tuned models still need improvement, gained valuable lessons and actionable observations for the next round of experiments.

AI for trading – potential methods

Considerations

- Data history
- Drivers' complexity
- Predictability
- Profitability
- Label balance

- Trading time horizon
- Trading philosophy
- Context
- Model complexity

- LLM selection
- Compute
- Data points

- Domain expertise
- Time & Cost
- State representation

Asset class selection

LLM-based

- Fine-tune
  - Expert labels
  - Enhanced state
  - Second fine-tune
  - Ensemble
- Agents
  - RAG
  - Fine-tuned
  - Ensemble

Machine Learning

- Reinforcement Learning
  - More compute, Reward design
- Transformers
  - More compute, Hyperparams
- LSTM
  - Hyperparams
- XGBoost
  - Hyperparams

Considerations

- Data history
- Drivers' complexity
- Predictability
- Profitability
- Label balance

- Trading time horizon
- Trading philosophy
- Context
- Model complexity

- Data points
- Directional prediction
- Reward design

- Grid Search
- Hyperparameters
- Noise reduction
- Label imbalance

* COMPLETED PROJECTS ARE IN GREEN IN THE CHART ABOVE.

Project summary: LLM fine-tuning pipeline

Asset class selection
- Crypto
- Market inefficiencies to exploit
- Relatively clear drivers
- Downsides: non-stationarity and short data history

Data preparation
- Technical indicators
- Market context
- Bitcoin leading indicators
- Rolling news summaries
- About 15k datapoints
- Target: price direction

LLM selection
- Mistral 7B
- Llama 3.1 8B
- Fin Llama 8B

Fine-tuning
- Google Colab
- PEFT - QLORA
- Cross-validation
- Class weighting
- F1: ~ 56%
- Likely underfitted

Testing
- Instruction following issues
- Output parsing issues
- Insufficient performance
- Regime recognition challenges

Lessons learned from LLM fine-tuning so far

🏷️ **Labeled data** should include all relevant drivers and, ideally, a brief rationale for the chosen action, so the LLM grasps nuances from inputs and learn whys. Providing rolling context can help LLMs better learn.

🧩 **Parsing** outputs from smaller LLMs can be tricky; their responses are often inconsistent or hard to structure.

⚠️ **Instruction following** is weaker in smaller LLMs (e.g., inventing summaries or ignoring explicit instructions).

🗣️ Free-text outputs often outperform classification heads, likely due to enhanced **reasoning capabilities**, even though probabilistic classifications are simpler to parse.

📑 Fine-tuning is more effective when inputs (e.g., technical indicators, news, leading indicators, macro metrics) are presented **with a narrative** or in a "report" format, rather than as raw data.

🌐 To improve robustness and generalization, it's crucial to provide **diverse examples** spanning various macro and market regimes, as the model struggles with generalization even when unseen market states are only moderately different from training data.

🚩 Fine-tuning, especially of already fine-tuned LLMs, can sometimes **diminish general understanding** capabilities. This can lead to issues such as failing to recognize clear "sell" indicators.