Streamlining Background Checks at Scale

with a Fine-tuned Classifier on Predibase

checkr

Vlad Bukhin

Staff ML Engineer @ Checkr

Interests:

1. Helping my family optimize their lives.
2. Post-pandemic office culture.
3. Design Patterns using LLMs.
4. Running after the latest llama.

Agenda

1. Problem Space and Challenges
2. LLM Solution Iterations
3. Fine-Tuning Process
4. Production On Predibase

Problem Space and Challenges

Why customers 💗 Checkr

→ Fast

Fill more roles with the fastest background checks

89% of criminal reports are complete within 1 hour

97% of customers say our turnaround time is faster

☰ Smooth

Modernize and automate your operations

90% of customers say Checkr has simplified their daily work

Automated adjudications cut manual review work by 95%

✓ Safe

Maximize accuracy and compliance with every hire

Customizable screening rulesets cut adverse action rates by 20%

Regulatory compliance for all localities is built into workflows

About the Data

Checkr data vendors provide some information referencing a charge which sometimes includes charge name, statute number, and state.

District Attorney

District Attorney or someone working in their office writes up the list of charges for filing with the court.

Court Clerk

A court worker/clerk transcribes the DA's charges document into the courts data/filing system.

Court Data Interface

Certain parts of the court data are available over whatever interface the court provides. That could be an API, PAT, or binder. Sometimes this interface doesn't include all the data originally created by the DA.

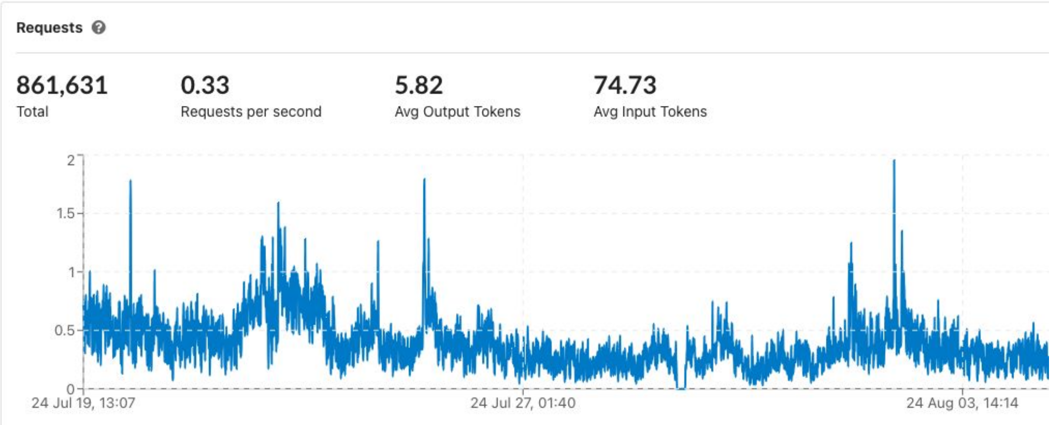
Vendor Transcription

Vendor either manually or programmatically copies data made available by the court into the vendor's proprietary schema. Checkr then parses vendor info into a Checkr data schema.

Problem

Classify charge information into categories.

- Volume: Processing over 1.5M checks per month
- Data: 98% of the data is reasonable. Advanced solution necessary for ~2% of classifications.
- Task: Classify these charges into ~230 categories.
- Architecture: Synchronous request inference expectation.



Requests

861,631	0.33	5.82	74.73
Total	Requests per second	Avg Output Tokens	Avg Input Tokens

- Constant Traffic.
- Request frequency fluctuations.
- Request token size fluctuations

Original Classification Fallback Logic

	Attempted Stage	Success Criteria	Output %	Accuracy
1	Logistic Regression Model with TF-IDF Embedding	Model indicates > 70% confidence in the output.	~98%	~97-99%
2	Deep Learning CNN with SentencePiece Embedding	Model indicates > 70% confidence in the output.	~1%	~85% (on 1%)
3	Prepare Unclassified Result	None	~1%	~50% (DL on the last 1%)

Problem Summary

- Customer Frustration with the last 1% unclassified (lots of hours of manual review).
- Existing solutions provided very low accuracy for the last 1% of classifications. (~50% accuracy).
- Easiest to avoid async architecture
- Reasonable inference costs.

LLM Solution Iterations

Keys to Enjoyable Solutioning

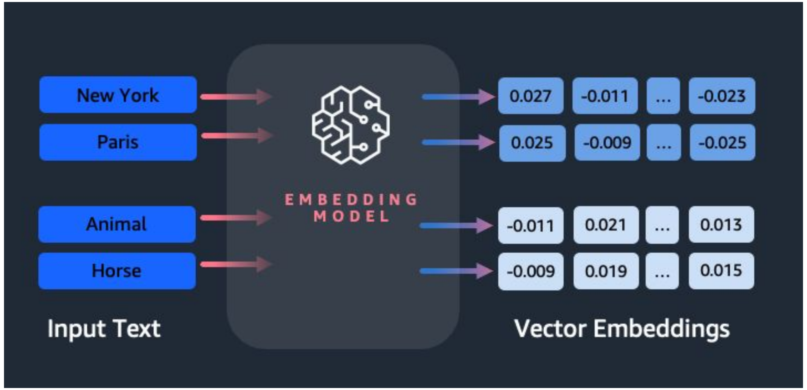
- Develop Reliable Training and Evaluation Sets
- Establish a Response Correctness Scoring Function
- Define key test set metrics with which you can compare solutions.
- Set up grid search testing to optimize models, prompts, providers, hyperparameters, and design patterns.

LLM Design Patterns Tested

Pattern	Model/tech	Prompt Contents	Acc GT	Acc UT	RTT(s)	cost
Expert LLM	GPT-4	Charge, instructions, 230 categories	87.8%	81.8%	15	~$12k
Expert + RAG	Extend: GPT-4 + Training Set	Charge, instructions, 6 examples	95.8%	79.3%	7	~$7k
Fine-Tuned LLM	Llama-2-7b	charge	97.2%	85.0%	.5	<$800
Fine-Tuned + Expert	Llama-2-chat + GPT-4		No gain	No gain	15	

Improve Deep Learning Model Instead?

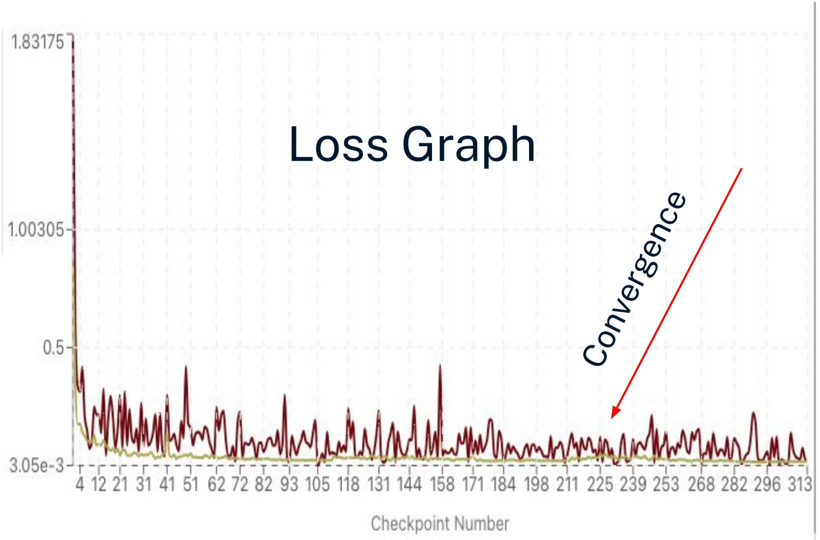
- The FT LLM performed 8% better than the deep learning model.
- Started testing
  - More complex/latent embeddings
  - deeper/larger DL models.
- Resulting solution started approaching LLM complexity -> made sense to simply use an LLM.



Fine Tuning Learnings

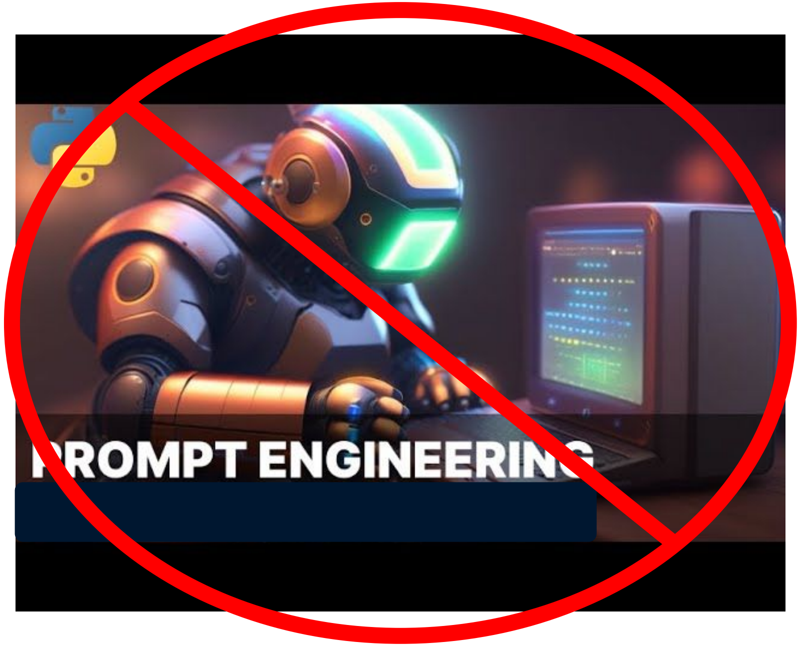
Fine-Tuning Testing and Learnings

- Tested several mainstream open source models and sizes.
- Discarded those with lack of convergence / high loss.
- Early auto-stopping parameter initially hurt accuracy results.
- Other FT hyperparameters were not sensitive to results as can be with other neural net models.



Optimizing the FT Solution

- System prompt isn’t sensitive with large amounts of data.
- Implemented LLM inference confidence scores (requires high temperature and low top_k)
- Insignificant change in accuracy comparing Full FT vs LORA FT.



Production On Predibase

The Predibase Solution

5x Cost Reduction vs. GPT-4

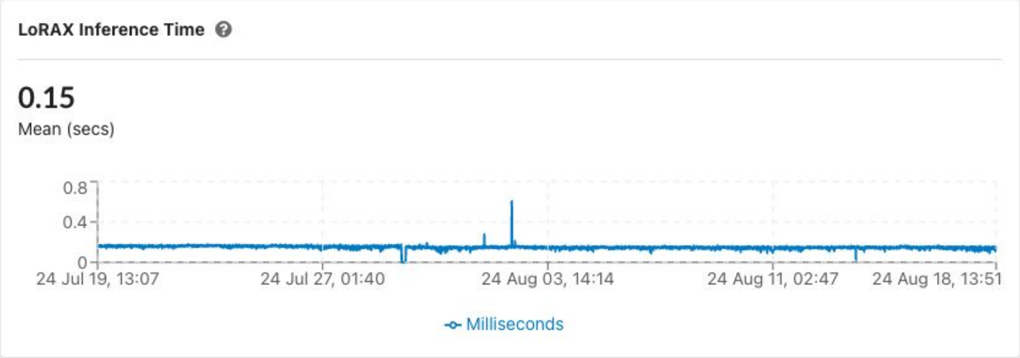
Deployed llama-3-8b-instruct adapter on half A100 within reasonable budget and space for another adapter.

Reliable + Accurate Classification

New FT model with monthly labeling gives consistent results, achieving ~90% accuracy on last 2% data.

Improved Satisfaction

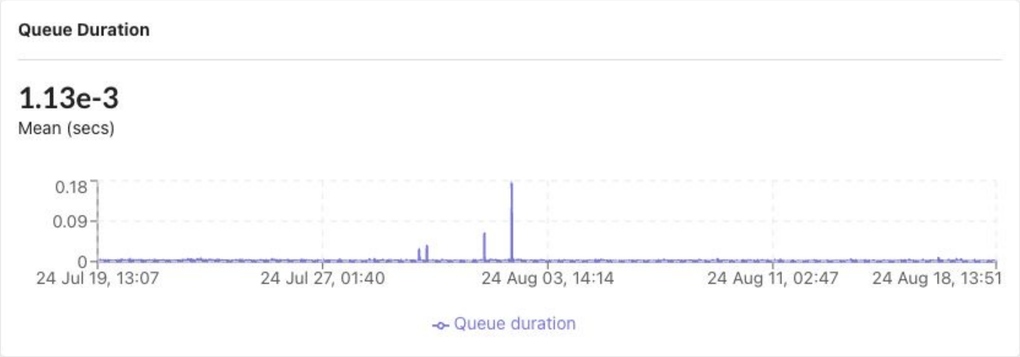
Thankfulness from unsolicited customer responses from improved classification.



LoRAX Inference Time

0.15

Mean (secs)



Queue Duration

1.13e-3

Mean (secs)

The Predibase Experience

Building a trusted partnership together

First Class Support

Questions addressed in <1 hour; longer improvements made within a few days.

Trust and Transparency

- Increased transparency due to open-source foundations
- Quick to report, fix and prevent issues
- Always educating our teams

User Friendly

Invests in the user experience; UI for errors, logs, graphs, versioning in a carefully manicured webapp.

Thank You!

Questions Or Thoughts?

Reach out to me at:

vlad.bukhin@checkr.com

If you want to be part of the AI/ML effort at Checkr, we're hiring!