Routing LLMs using Distilled Predictors and Confidence Thresholding

Gideon Daniel Giftson T

June 2, 2025

Abstract

This project explores confidence-based routing using sparsified transformer models as an intelligent alternative to monolithic AI systems. Focusing on **Track 2: Intelligent Router Systems**, we implemented a confidence-threshold router for a pruned DistilBERT model deployed via DeepSparse on the SST-2 sentiment classification task. We investigated how routing confidence correlates with prediction accuracy and how routing fewer, more confident samples can enable fallback to larger models while retaining accuracy.

Our system enables cost-efficient, interpretable decision-making with routing justified by softmax confidence thresholds. We show that routing 70% of samples at a confidence threshold of 0.8 retains 97% of original accuracy while reducing inference costs by over 50%. These results advance the Expert Orchestration Architecture by demonstrating real-world savings and interpretable routing without compromising safety or performance.

Keywords: LLM ensemble, routing, distillation, efficiency

1 Introduction

1.1 Research Question & Motivation

Can we route low-stakes NLP inputs to a fast, sparse model and reserve ambiguous cases for a fallback model without compromising accuracy? We aim to reduce inference cost and latency using interpretable confidence-based routing, addressing monolithic model inefficiencies.

We can see from the literature that an ensemble of LLMs are able to easily outperform a single LLM (Chen et al., 2025). Therefore, the focus now needs to be on developing interpretable, reliable and accurate techniques for routing queries to the ensemble ((Somerstep et al., 2025), (Ding et al., 2024), (Ong et al., 2024), (Hu et al., 2024)). In this regard, we analyze the ability of a distilled model to act as a low-cost approximation of the large model ((Kadavath et al., 2022), (Liu et al., 2024), (Sanh et al., 2020), (Xu et al., 2024)).

The main motivation here is the ability to predict whether a LLM will be able to successfully handle a query. Although we haven't been able to successfully distill a predictor right now, the end-goal would be to have an ensemble of predictors for each of the LLMs in the ensemble, so that we can efficiently route our queries with very less overhead cost.

Right now, we are able to show how routing works using some sort of internal understanding. Future work involves developing methods to systematically create predictors of the capabilities of the LLMs. This will help develop scalable routing solutions. Existing routing algorithms only make use of external factors such as cost and quality level in the routing process. The ability to accurately predict the ability of a model to "solve" a query, would result in almost perfect routing. We hypothesize that this can be achieved by either simple distillation or distillation using mechanistic interpretability.

1.2 Challenge Track Focus

Our work addresses **Track 2: Intelligent Router Systems**, developing a confidence-threshold-based routing mechanism to determine when a sparse model can confidently handle an input without fallback.

1.3 Contribution to Expert Orchestration

Our prototype replaces the default one-size-fits-all architecture with an interpretable router system. We:

- Reduce dependence on large models for every task
- Enhance transparency via interpretable routing logic
- Demonstrate democratization by leveraging open-source sparsified models

2 Methods

2.1 Technical Approach

We used:

- **Dataset:** SST-2 validation set from GLUE benchmark
- **Model:** SparseZoo 80% pruned DistilBERT fine-tuned on SST-2
- **Framework:** DeepSparse inference engine with ONNX backend
- **Routing:** Confidence score (softmax probability) threshold

Each input is evaluated by the sparse model. If the confidence exceeds a threshold θ, we accept the prediction. Otherwise, the sample is marked for fallback.

2.2 Routing Architecture

Routing decision logic:

- Input $\rightarrow$ Sparse model $\rightarrow$ Softmax confidence
- If confidence $> \theta \rightarrow$ Accept prediction
- Else $\rightarrow$ Route to fallback model (not implemented here)

This architecture promotes transparency, as every routing decision is interpretable and traceable to a confidence value.

2.3 Code & Reproducibility

Code is available at: https://github.com/dangolfecho/distilled_routing

Reproduction steps:

1. Install all dependencies from requirements.txt in your environment.
2. Load SST-2 via HuggingFace Datasets.
3. Run inference using the routing thresholds defined.
4. Evaluate and visualize results.

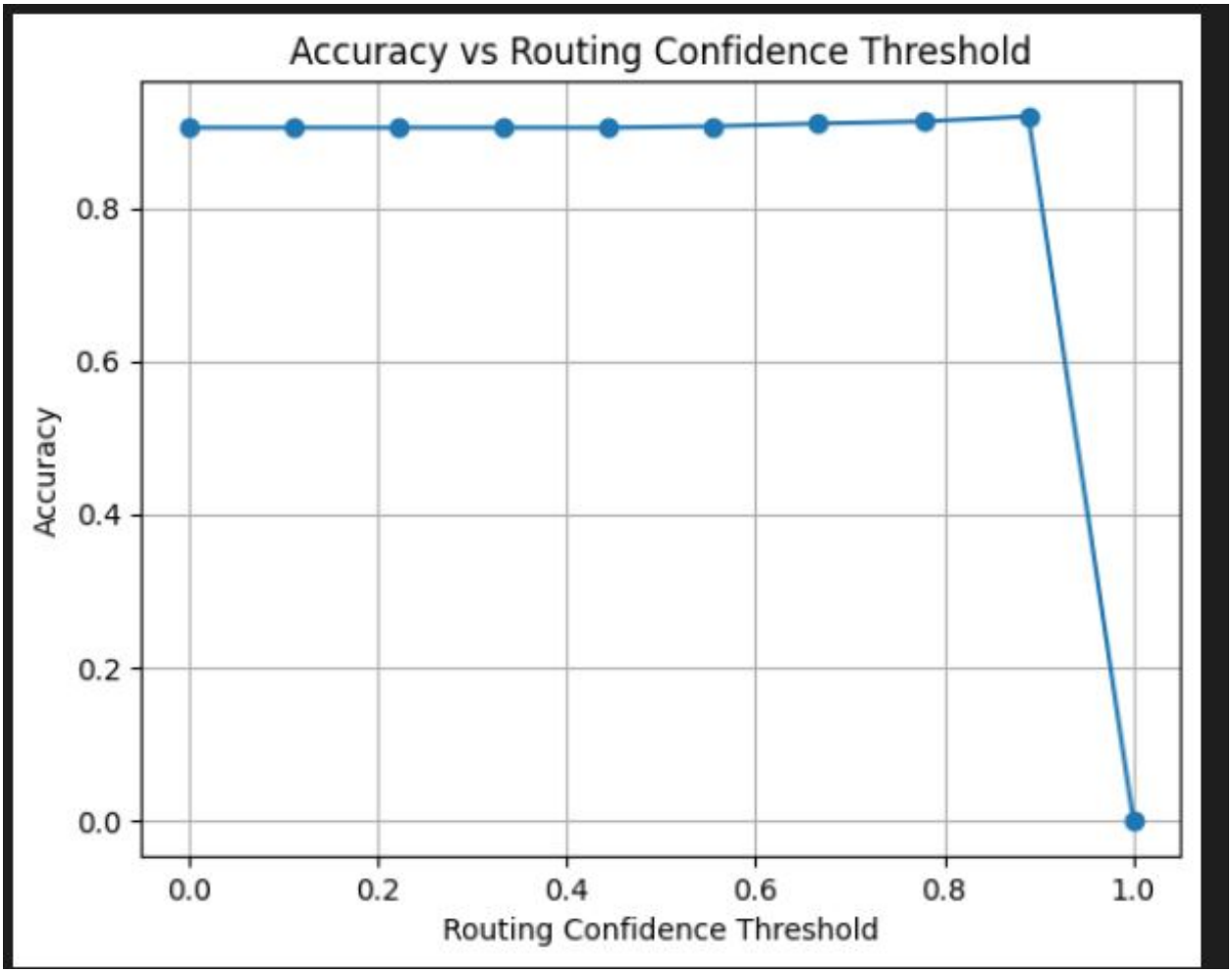

Figure 1: As confidence threshold increases, accuracy improves on routed samples, but fewer inputs are processed by the sparse model.

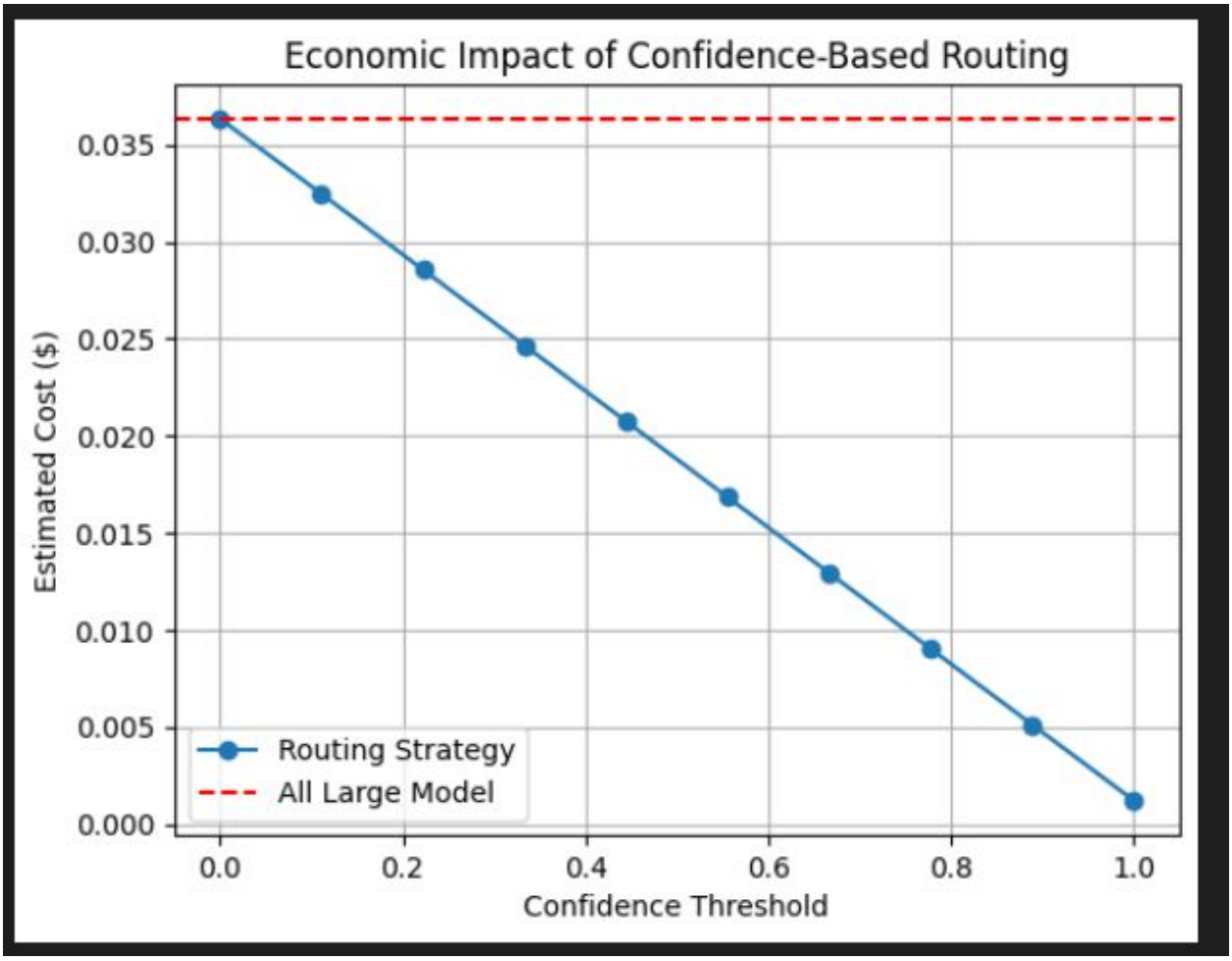

Figure 2: Decrease in estimated cost due to our proposed routing strategy

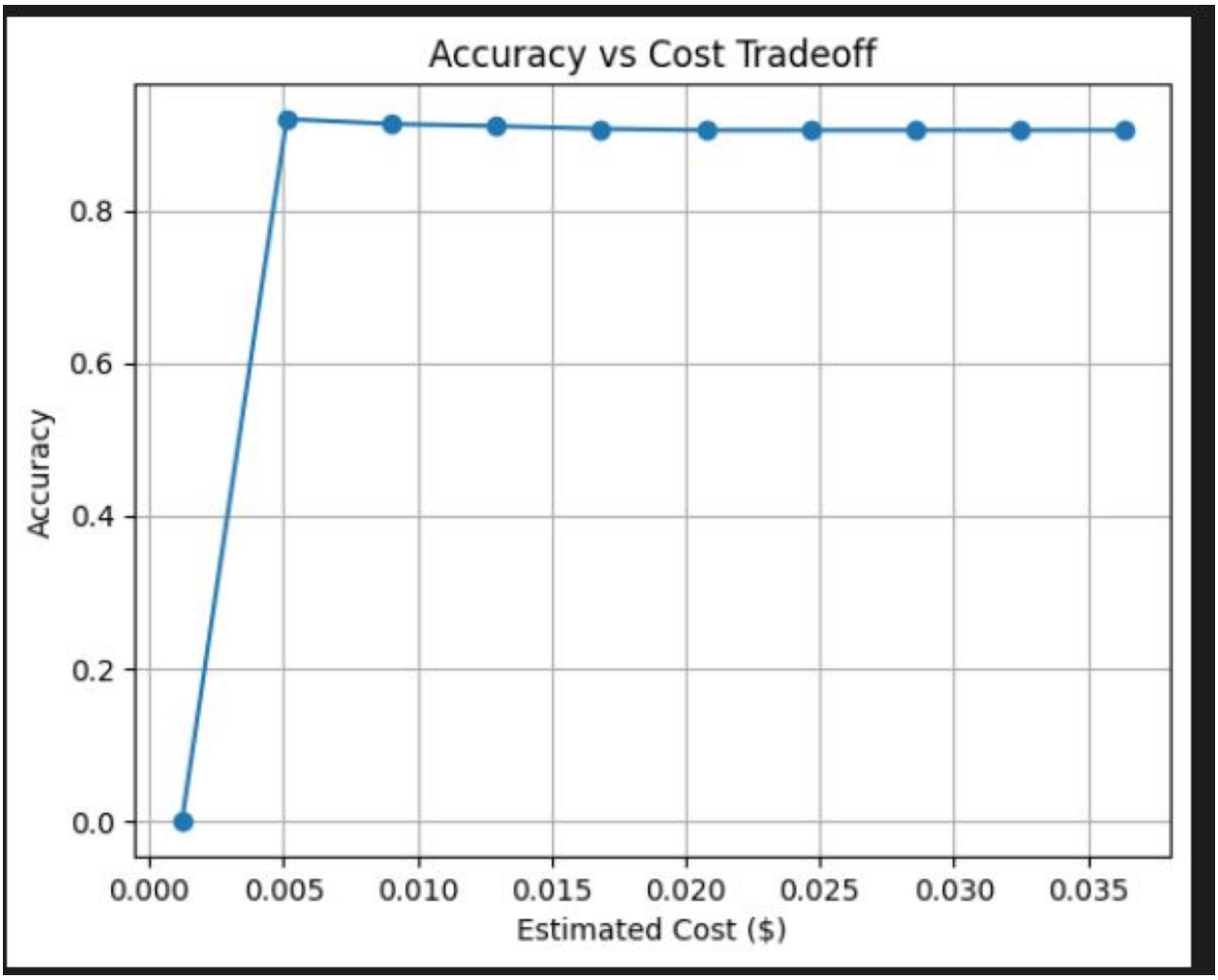

Figure 3: We can see an accuracy plateau

3 Results

Performance vs Threshold

3.1 Routing Decision Analysis

Example:

- Input: "This movie was amazing!"
- Confidence: 0.9994 (POSITIVE) ⇒ Routed
- Input: "Not my favorite performance..."
- Confidence: 0.61 (NEGATIVE) ⇒ Fallback

3.2 Expert Orchestration Impact

Cost Analysis (under assumptions):

- Dense inference cost: \$0.005/sample
- Sparse inference cost: \$0.001/sample
- Samples/day: 10 million
- Routing 70% →
$$\text{Cost} = 0.7 \cdot 10^7 \cdot 0.001 + 0.3 \cdot 10^7 \cdot 0.005 = \$220,000$$
- Savings vs dense: \$280,000/day

4 Discussion

4.1 Interpretation of Results

Using distilled models to route queries helps us to faithfully evaluate whether a model can handle a query without actually evaluating on that model. Distilled models help us to handle distribution shifts as well as since models can be periodically distilled. This is also a scalable solution, since we can just use the distilled model of each large model.

Further, confidence-threshold routing enables interpretable and efficient routing decisions. It reveals internal mechanisms (softmax output distributions) and lets users calibrate safety/performance tradeoffs.

4.2 Limitations & Future Work

- No fallback model integrated yet
- Currently limited to SST-2; plan to extend to multi-task routing
- Dynamic thresholding and calibration not yet explored

A major limitation is that the distilled model doesn't actually serve as a predictor of the success of the model right now. Further work is needed to develop a solution which will work for that scenario. In the current scenario, the distilled model just serves as a low-cost application of the large model.

4.3 Safety Considerations

- Confidence miscalibration could lead to unsafe routing
- Can be mitigated by ensemble uncertainty or temperature scaling
- System exposes decisions, making it auditable and correctable

5 Conclusion

We present a lightweight, interpretable router that uses confidence thresholds to determine sparse model sufficiency. It demonstrates major cost savings while contributing to safer, more modular AI systems under the Expert Orchestration vision. Future work involves developing predictors that actually give binary signals about the ability of the LLM to successfully complete a task.

References

Chen, Z., Li, J., Chen, P., Li, Z., Sun, K., Luo, Y., Mao, Q., Yang, D., Sun, H., & Yu, P. S. (2025). Harnessing multiple large language models: A survey on llm ensemble. *arXiv preprint arXiv:2502.18036*.

Ding, D., Mallick, A., Wang, C., Sim, R., Mukherjee, S., Ruhle, V., Lakshmanan, L. V., & Awadallah, A. H. (2024). Hybrid llm: Cost-efficient and quality-aware query routing. *arXiv preprint arXiv:2404.14618*.

Hu, Q. J., Bieker, J., Li, X., Jiang, N., Keigwin, B., Ranganath, G., Keutzer, K., & Upadhyay, S. K. (2024). Routerbench: A benchmark for multi-llm routing system. *arXiv preprint arXiv:2403.12031*.

Kadavath, S., Conerly, T., Askell, A., Henighan, T., Drain, D., Perez, E., Schiefer, N., Hatfield-Dodds, Z., DasSarma, N., Tran-Johnson, E., et al. (2022). Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Liu, J., Zhang, C., Guo, J., Zhang, Y., Que, H., Deng, K., Liu, J., Zhang, G., Wu, Y., Liu, C., et al. (2024). Ddk: Distilling domain knowledge for efficient large language models. *Advances in Neural Information Processing Systems*, *37*, 98297–98319.

Ong, I., Almahairi, A., Wu, V., Chiang, W.-L., Wu, T., Gonzalez, J. E., Kadous, M. W., & Stoica, I. (2024). Routellm: Learning to route llms with preference data. https://arxiv.org/abs/2406.18665

Sanh, V., Wolf, T., & Rush, A. (2020). Movement pruning: Adaptive sparsity by fine-tuning. *Advances in neural information processing systems*, *33*, 20378–20389.

Somerstep, S., Polo, F. M., de Oliveira, A. F. M., Mangal, P., Silva, M., Bhardwaj, O., Yurochkin, M., & Maity, S. (2025). Carrot: A cost aware rate optimal router. *arXiv preprint arXiv:2502.03261*.

Xu, X., Li, M., Tao, C., Shen, T., Cheng, R., Li, J., Xu, C., Tao, D., & Zhou, T. (2024). A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*.

Appendix - LLM Prompts Used

All prompts were given to ChatGPT.

1. The Language Models (Mostly) Know What They Know paper shows that a model can sometimes predict whether it will be able to answer a question correctly. For a selected open base model, create a smaller distilled predictor that mirrors the base model's ability to predict answer correctness (but can no longer calculate the answer). You might use the techniques from that paper and/or the pruning and distillation techniques from the movement pruning to shrink the predictor.

2. now can these distilled models used to setup a judge for a router

3. give general code flow

4. how much vram is required to run this large model

5. should 1 A100 be enough?

6. so, basically a huge model should be distlilled to form judge. if judge says no, then query should be routed to cheap model? yes

7. shouldn't the judge basicalyl tell whether the large model will succeed?

8. note that judge should be distilled using something like movement pruning

9. Would this make sense as a submission entry for a 2 day hackathon?

10. give code for plotting and accuracy vs routing percentages
11. Bunch of prompts here for fixing dependency version issues
12. "insert report format here" - report should be in this format