Reducing Environmental Costs whilst Maintaining Operational Effectiveness in Large Language Models through Chain Routing

Solomon Wheeler[1], **Melissa Torgbi**[1], **Harish Tayyar Madabushi**[1]
[1]Department of Computer Science, University of Bath
{sollywheeler@gmail.com, mat66@bath.ac.uk, htm43@bath.ac.uk}

Abstract

The widespread deployment of large language models (LLMs) presents significant challenges in balancing their performance and carbon cost. To address this, we propose a prompt routing mechanism that balances quality and environmental cost to route prompts between a set of candidate LLMs. In addition to fine-tuning encoder models to account for carbon cost, we introduce *chain routing*, a novel routing mechanism that breaks responses into three stages, which can be routed to different LLMs. Our system reduces environmental cost by 39% whilst simultaneously improving response quality by 3.3%, both compared to the single highest performing model. We also show that chain routing outperforms the quality of single model routing by 3.1%, whilst only having a slightly higher environmental cost. These results show the potential for environmentally aware routing, and *chain routing* to improve the environmental sustainability of LLM inference. We provide open access to our code [1].

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities, exceeding human accuracy on some benchmarks (Annepaka and Pakray, 2025; Bojic et al., 2023; Goldstein et al., 2023). However, they are not without their drawbacks, specifically cost and environmental impact (Mittelstadt et al., 2023; Luccioni et al., 2024; Samsi et al., 2023). The rise of reasoning models, such as Deepseek's R1 and OpenAI's o1/3, has exacerbated these issues; to reason before answering, they require significantly more compute per prompt (Han et al., 2025; DeepSeek-AI et al., 2025; OpenAI, 2024).

Whilst existing work highlights the potential for routing to improve accuracy and reduce cost (Stripelis et al., 2024; Shnitzer et al., 2023; Damani et al., 2024), environmental impact is an area highlighted for further research (Hu et al., 2024; Varangot-Reille et al., 2025). Wilkins et al. (2025) factor environmental cost into routing, but this routing is offline and does not consider prompt context. To address this gap, we introduce an online environmentally aware router that considers prompt context in routing decisions.

Furthermore, whilst previous approaches have explored swapping LLMs during inference, this is done at the token level with a limited set of models (Zheng et al., 2024). We propose a more structured approach, chain routing, which breaks Chain-of-Thought (CoT) responses into three distinct stages. Each stage can be routed to a different candidate LLM, balancing performance and carbon cost throughout the reasoning process, as shown in Fig 1.

Our key contributions are:

1. We create a system that predicts the quality and environmental cost of LLM inference for a given prompt and uses this to make routing decisions. To the best of our knowledge, this is the first time this has been done.

2. We measure the quality and environmental cost of inference and training across a range of routing methods.

3. We create a novel routing system, *chain routing*, which routes different parts of a reasoning CoT to different LLMs. We also compare the quality of smaller and larger LLMs for each stage of a reasoning CoT.

In comparison to the highest performing candidate LLM, our router reduces the environmental costs of inference by 39% whilst also increasing quality, measured in similarity to gold standard responses, by 3.3%. We also show that chain routing outperforms single model routing by 3.1%, whilst only having a slightly higher environmental cost.

[1]Code for our routing system can be found at https://github.com/solomon-wheeler/ChainRouterToReduceCarbonCosts

Proceedings of the 21st Conference on Natural Language Processing (KONVENS), Volume 2: Workshops, pages 250–273, 2025

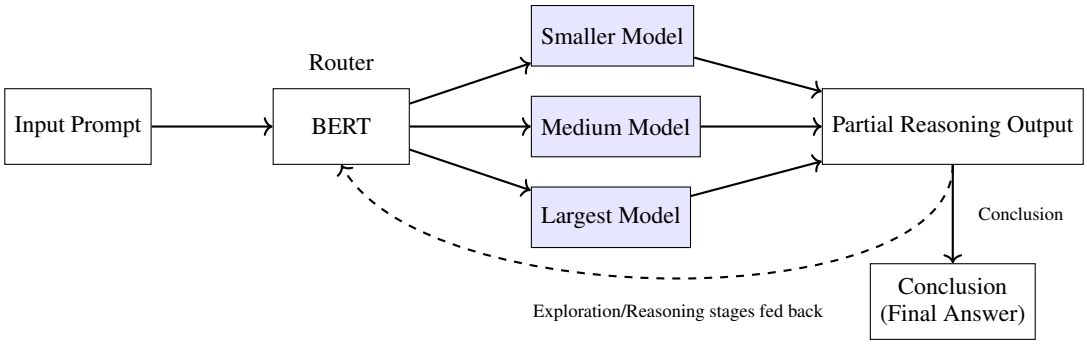

Figure 1: Overview of the chain routing system, which splits a CoT output into discrete stages, and routes these partial reasoning outputs to different models, before outputting a final answer.

Our results also show that there is little impact on quality when using smaller models ($3B$ parameters) instead of larger models ($32B$ parameters) for the initial two stages of reasoning (both less than 2.4% reduction in quality), a finding that could be used to decrease the environmental cost of CoT reasoning in LLMs.

2 Background & Related Work

2.1 LLM Routing

LLM routing is an area of intense research; several methods have been proposed to decide which of a set of candidate LLMs will best respond to a prompt. These methods include training encoder-only models, typically from the BERT family, to classify prompts based on factors such as domain relevance (Yadav et al., 2025; Simonds et al., 2024) or anticipated performance (Stripelis et al., 2024; Hu et al., 2024; Ding et al., 2024). Decoder-based models have also been employed for similar classification tasks (Stripelis et al., 2024; Ong et al., 2024; Shen et al., 2023). Alternative approaches include clustering semantically similar prompts using KNNs or K-means clustering and selecting the model that performed best on similar prompts (Srivatsa et al., 2024; Stripelis et al., 2024; Hu et al., 2024). Finally, reinforcement learning (RL) approaches learn routing policies over time (Zhang et al., 2024b; Nguyen et al., 2024)

2.2 Alternative Approaches

LLM Cascading Instead of routing before output is generated, cascading approaches send prompts to the smallest model and then push them up a stack of increasingly powerful models until the output is high quality. This measure of quality differs between systems; one method is to take the uncertainty of a model's output, either by measuring the likelihood of the tokens (perplexity) (Zhang et al., 2024a), or by running inference multiple times and comparing how similar the outputs are (Yue et al., 2023; Aggarwal et al., 2024). Another method is to use fine-tuned encoder models to classify if outputs are high quality (Chen et al., 2023). Cascading systems must measure response quality online, during inference. Since we use a routing system that only needs quality scores offline to generate training labels, we can use a simpler method of similarity between gold standard answers and responses as a surrogate for quality.

Mixture of Experts LLM routing systems are similar to the mixture of experts (MoE) architecture, which includes several expert sub-models (Yadav et al., 2025; Jacobs et al., 1991; Cai et al., 2025). LLMs using this architecture include Mistral AI's Mixtral 8×7B model, DeepSeek's R1, and Meta's Llama-4 family (Jiang et al., 2024; DeepSeek-AI et al., 2025; Meta, 2025).

The key difference between LLM routing and MoE systems is how tightly coupled model selection is to models. In MoE systems, the gating method is trained with the sub-models, so updating any sub-model necessitates updating the entire system (Shazeer et al., 2017). In contrast, LLM routing systems train the gating/routing system separately, so can include new models more easily (Jiang et al., 2023).

2.3 Environmental Impact of LLM Inference

We show estimates from previous work of the environmental impact of LLM inference in Table 1.

Model	Carbon Cost (g CO_2)	Citation
Llama 65B	0.4096	Samsi et al. (2023)
Llama 2 13B	0.21	Li et al. (2024)
Llama2 70B	1.14	Wilkins et al. (2025)
GPT-3.5	1.869	d'Aramon et al. (2024)

Table 1: The carbon cost of LLM inference, normalised to a carbon intensity of $400g$ CO_2/kWh and output length of 1024 tokens.

3 Methodology

3.1 System Overview

Our primary routing system focuses on a fine-tuned BERT classifier, which takes input of the prompt (and chain output so far, if applicable), and outputs soft scores, similar to the method proposed by Stripelis et al. (2024). We update the BERT classifier to output two soft scores, for the expected quality and environmental cost. We also include a decoder router as a baseline, which employs a smaller LLM to make routing decisions. The system operates in two modes: single routing, in which the entire prompt is sent to one model, and chain routing, in which different stages of a reasoning CoT are delegated to different models.

3.2 Routing Architectures

Each prompt is tokenised and then encoded using a model from the BERT family. The training data for this was gathered by sending prompts from several datasets to each candidate LLM and recording the similarity between the output and gold standard answer, and the environmental cost of inference. For gathering the chain routing training data we also instruct the model to output its CoT.

3.3 Problem Formulation

In both chain and single routing, for every prompt $x \in X$ and candidate LLM $m_i \in M$, the router predicts two soft scores:

$$\text{Sim}_i(x) \in [0,1], \qquad \text{Env}_i(x) \in [0,1], \quad (1)$$

Sim_i and Env_i are the soft scores for expected similarity to the gold standard answer and environmental cost, the calculation of which is explained in the next section.

We then calculate a combined score for each candidate LLM, including $\lambda \in [0,1]$, which varies the trade-off between expected similarity and environmental cost, with lower values λ favouring increasing similarity:

$$\begin{aligned}\text{CombinedScore}_i(x) = (1-\lambda)\,\text{Sim}_i(x) \\ +\lambda\,(1-\text{Env}_i(x)).\end{aligned} \quad (2)$$

At inference time, the router then selects the candidate LLM with the highest score:

$$m^*(x) = \underset{m_i \in M}{\text{argmax}}\ \text{CombinedScore}_i(x). \quad (3)$$

3.4 Soft Score Calculation

We expand on the soft score implemented by Stripelis et al. (2024) by predicting two soft scores for each candidate LLM: the similarity score (Sim_i) and environmental cost (Env_i). The general equation for this soft score (ϕ_i) is

$$\phi_i(S;T) = \frac{\exp\left(\frac{s_i}{T}\right)}{\sum_{j=1}^{N}\exp\left(\frac{s_j}{T}\right)} \quad (4)$$

where N is the number of models, S is the output score, and s is the input score. The temperature (T) controls the sensitivity of the probability distribution. As T tends towards infinity, the output tends towards a uniform distribution, and as T tends towards 0 the output is 1 for the highest value, and 0 for all other values (Xuan et al., 2025). The input score is the calculated similarity score for Sim_i or the environmental cost for Env_i.

For chain routing, we require a soft score for each model at each CoT stage. For example, in the first CoT stage, we must calculate a soft score for each of the three candidate models, even though additional routing decisions will be made for the subsequent CoT stages. To address this, we compute the average soft score across all possible model paths extending from the current stage ($\bar{\phi}_i$). For each model at CoT stage i, we take the average soft scores of each of the remaining model paths P_i:

$$\bar{\phi}_i = \frac{1}{|P_i|}\sum_{j \in P_i}\phi_j \quad (5)$$

3.5 Reasoning Chain Decomposition

To route different stages of a reasoning CoT to different candidate LLMs in chain routing, we require

a method to split a CoT into discrete stages. We split the reasoning steps into three distinct sections, **exploration**, **reasoning** and **conclusion**, loosely following the special tokens used in DeepSeek-R1 (DeepSeek-AI et al., 2025). We instruct the current model to generate a CoT stage and terminate it with a predefined token (e.g. "EXPLORATION"). Once the model outputs this token, we stop decoding, and pass the prompt and output so far to the router to select the LLM for the next CoT stage. The system prompt and an example CoT split into stages are shown in Appendices D.14 and D.15.

3.6 Implementation Details

Models are loaded in Python using the weights available on Hugging Face and run on A5000/3090 24GB GPUs. A full list of hyperparameters used can be found in Appendix A, and an example system prompt can be found in Appendix D.13.

Hyperparameters for training the BERT router are shown in Appendix B.1. We use KL-divergence loss with weighted classes to reduce the effect of imbalanced classes, the formula for this can be found in Appendix B.2.

4 Experimental setup

4.1 LLMs Used for Router Training and Inference

To train the BERT router, we first require labels for the similarity score and emissions output for prompts. We therefore send a subset of prompts from each dataset to a broad selection of LLMs, selected due to their high performance-to-parameter size ratio, to maximise performance, whilst minimising environmental cost. We also select smaller task-specific models, such as Deepseek Coder, which performs well in the programming domain, whilst only having 1.6 billion active parameters (DeepSeek-AI et al., 2024). The candidate LLMs we use can be found in Table 2.

Model family	Params (B)
Llama 3.2(Grattafiori et al., 2024)	1
	3
Llama 3.1(Grattafiori et al., 2024)	8
DeepSeek Coder-v2(DeepSeek-AI et al., 2024)	16
DeepSeek R1-Distill(DeepSeek-AI et al., 2025)	1.5
	32
Qwen 2.5(Qwen et al., 2025)	32

Table 2: LLM family and parameter sizes used both for gathering labels for training the BERT router and as candidate models for selection in single routing.

4.2 Chain Routing Models

Running for all model combinations for each CoT stage is intractable due to the exponentially increasing number of combinations, so the set of models is refined for the chain router. Distilled reasoning models are excluded, as we find their predefined reasoning format precludes them from responding with a custom structure without further fine-tuning. Therefore, we select the following models to run for all combinations on the CoT dataset:

1. Llama 3.2 3B (Grattafiori et al., 2024)
2. Llama 3.1 8B (Grattafiori et al., 2024)
3. Qwen 2.5 32B (Qwen et al., 2025)

4.3 Encoder Models

We evaluate the following encoder models at routing:

1. DeBERTa-v3 base 183M (He et al., 2023)
2. RoBERTa base 125M (Zhuang et al., 2021)

We select DeBERTa-v3 as it achieves best-in-class performance in natural language understanding tasks and is used in NVIDIA's routing blueprint (He et al., 2023; NVIDIA, 2025). RoBERTa is selected as previous work found it outperforms DeBERTa for routing (Feng et al., 2024; Srivatsa et al., 2024).

4.4 Datasets

Datasets are selected from the three types of reasoning benchmarks outlined by Wang et al. (2023): arithmetic, common sense and symbolic. To create a generalisable router, we also include MMLU (Massive Multitask Language Understanding) and MMLU-pro, which have questions across 57 topics. Coding prompts from the MBPP (Mostly Basic Programming Problems) dataset are also included, as programming is commonly included in other routing systems (Stripelis et al., 2024; Hu et al., 2024). The final domain of datasets is question answering, which is included as it is a good differentiator between model performance (Fischer et al., 2024).

Subsets of these datasets are combined for single routing, generally of between $300 - 500$ prompts, except for the datasets combining multiple tasks, such as Unified-QA and MMLU, which have between $1500 - 2500$ prompts to ensure each task

has an appropriate number of prompts, creating a dataset of over $9,000$ prompts for each model.

We reduce the number of prompts for chain routing to 200, and only use UNIFIEDQA from the question-answering datasets, creating more than $100,000$ soft scores.

Task	Dataset/s
Arithmetic reasoning	GSM8K (Cobbe et al., 2021)
Common sense reasoning	ARC (Clark et al., 2018)
Symbolic reasoning	GSM-Symbolic (Mirzadeh et al., 2024)
Multitask reasoning	MMLU (Hendrycks et al., 2021) MMLU-PRO (Wang et al., 2024)
Programming	MBPP (Austin et al., 2021)
Reading comprehension/ Question answering	UNIFIEDQA (Khashabi et al., 2020) SQUAD (Rajpurkar et al., 2016), GPQA (Rein et al., 2024)

Table 3: The benchmark datasets we use for measuring carbon cost and quality of responses for each model.

4.5 Baselines

We use the following baselines to compare the performance of the BERT router implemented:

- A generative (decoder) router (Llama 3.2 1B).
- The single model with the highest average similarity score.
- The single model with the lowest average emissions.
- A random router.
- An oracle router as a topline, which is 100% accurate at selecting the best model for a prompt, following the method by Hu et al. (2024).

4.6 Evaluation Metrics

We aim to create a system that reduces the environmental impact of LLM inference whilst maintaining accuracy. To quantify our success, we measure the environmental impact of inference (g CO_2) and the similarity between responses and gold standard answers, as a proxy for quality.

4.7 Similarity Comparisons

As a surrogate measure for the quality of responses, we measure the similarity between the outputs from models and gold standard responses defined in the benchmark datasets we use. Two methods are used for similarity comparisons: BERTSim score and CHRF (character n-gram F-Score) (Zhang et al., 2020; Popović, 2015). We instruct the models to output #### followed by their final answer and only calculate the similarity to the gold standard answer after this, to ensure we do not penalise reasoning models, which output reasoning tokens before their final answer.

4.7.1 BERTScore

We select BERTScore for measuring similarity between outputs and gold standard answers, as this approach correlates well with human judgements (Zhang et al., 2020). BERTScore encodes texts, computes token-wise cosine similarities, and aggregates them using mean, precision, recall, or F1. We simplify this by encoding the full sentences and taking a single cosine similarity, providing a more lightweight approximation. The BERTscore equation is

$$\text{BERTScore}(m, g) = \frac{\mathbf{e}_m \cdot \mathbf{e}_g}{\|\mathbf{e}_m\| \, \|\mathbf{e}_g\|} \tag{6}$$

where m is a given model's output, g the gold standard response, and e_m and e_g are their respective BERT encodings.

4.7.2 CHRF

We evaluated coding tasks using CHRF, as Evtikhiev et al. (2023) found this to be similar to human evaluation for code. CHRF is calculated as

$$\text{CHRF} = \frac{(1+\beta^2)\, M\, G}{(\beta^2 M + G) \times 100} \tag{7}$$

where M and G are the mean n-gram recall between output and gold standard, and gold standard and output, respectively, and β is a weighting parameter. We set β to 2, the value suggested by Popović (2016). We implement CHRF using the Evaluate library [2], and scale values to between 0 and 1 to match BERT score.

4.8 Calculating Carbon Cost

We calculate carbon cost using the CodeCarbon library (Courty et al., 2024), which works by recording the power consumption of CPU, GPU, and

[2] https://github.com/huggingface/evaluate

RAM and multiplying this by local carbon intensity[3], as shown in the equation:

$$\mathrm{CO_2\ (g)} = \frac{\text{power (kW)} \times \text{time (h)}}{1000} \times \text{Carbon Intensity}_{\mathrm{kg/kWh}}. \quad (8)$$

Details of the energy mix and hardware we use can be found in Appendix C. We use quantised versions of the largest models due to hardware limitations (hyperparameters can be found in Appendix A). To allow for fair comparison, we run 50 prompts on the unquantised version of each model and scale the recorded emissions to match these.

5 Experiments & Results

5.1 Environmental Cost

The environmental cost for each model averaged across all the datasets listed in Table 3 is shown in Figure 2. These results show two patterns in the data: firstly that larger parameter count generally leads to a larger cost of inference, and secondly, that reasoning models have a higher carbon cost than non-reasoning models.

We include a reasoning distilled and non-reasoning version of the same model (Qwen 2.5 32B) for comparison and find that the reasoning distilled version has a 35% higher environmental cost. This can be explained by DeepSeek's distilled version of Qwen-2.5 32B producing three times more output tokens on average in our experiments.

On average, our measured carbon costs are slightly higher than for the models in Section 2.3. This can be explained by the fact that these works employ much higher batch sizes (64 vs. 8), run on more powerful GPUs (A100 80GB vs. A5000/3090 24GB), and set much lower token limits (256 vs. 2500) (Samsi et al., 2023).

The environmental cost of gathering router training data, and training the single and chain routers is shown in Appendices D.2 and D.3. We find the largest carbon cost of router training is gathering a dataset of responses to train the router on. We ensured we minimised this cost as far as possible by only running this inference once, with only the models needed, and using a subsection of datasets to gather enough prompts to train the router, but not more.

[3] https://github.com/mlco2/codecarbon/blob/master/codecarbon/core/emissions.py

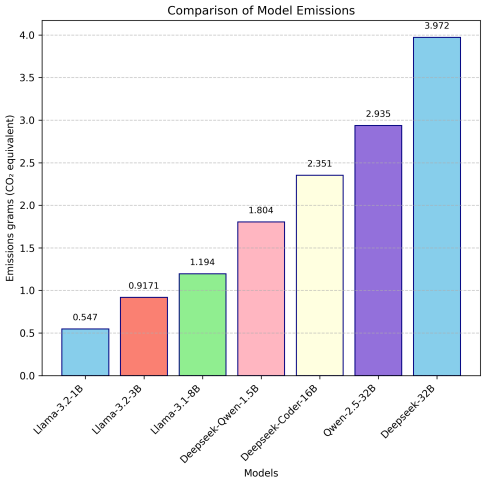

Figure 2: A comparison of the CO_2 emissions cost measured, averaged across all datasets.

Single routing BERT (DeBERTA-v3) has the lowest carbon cost (0.0001565 g CO_2) of our routing approaches. BERT in chain routing is higher (0.04102 g CO_2), as it is invoked once for each stage, with a growing input context each time. The carbon cost and latency of all implemented routing approaches can be found in Appendix D.1; we add the cost of routing to all results to obtain an overall system carbon cost.

5.2 Hyperparameter Selection

We evaluate the base version of two different encoder models, RoBERTa and DeBERTa-v3 (Zhuang et al., 2021; He et al., 2023), fine-tuned to predict soft scores. DeBERTa-v3 achieves a slightly lower average error (0.4491% vs 0.4401%), although it has a slightly longer average inference time (0.0976s vs 0.0619s), which can be explained by DeBERTa-v3's higher parameter count (183 million vs. 125 million) (He et al., 2023; Zhuang et al., 2021). We select DeBERTa-v3 to be used as the BERT model for the remaining results, as the increased inference time is negligible in comparison to the candidate LLMs.

The accuracy of the BERT router (DeBERTa-v3) in single routing at predicting both similarity and emissions on a dataset not used in training are shown in Table 4.

Routing Method	Top-1 Accuracy	Top-3 Accuracy
BERT (Similarity)	31.0%	71.4%
BERT (Emissions)	37.6%	85.2%
Random	14.3%	42.9%

Table 4: Single model router accuracy at predicting the highest quality/lowest emissions from the seven candidate models for a given prompt from an unseen dataset.

We also test varying λ, the trade-off between emissions and similarity scores introduced in Section 3.3. We model this trade-off as a Pareto optimisation problem and find $\lambda = 0.7$ to be the best. Emissions decrease significantly beyond $\lambda = 0.7$, but with a significant reduction in output quality. At $\lambda = 0$ the two most carbon-intensive candidate models handle the majority (57.3%) of requests, while the two most efficient models receive just 15.8%. Conversely, at $\lambda = 1$ only 16.2% of calls are sent to the two high-carbon models, and 26.5% to the two low-carbon models. Complete emission and quality values for all λ values, as well as how the usage of candidate models varies across λ values, are shown in Appendix D.4.

5.3 Single Model Routing

We evaluate single routing on the fine-tuned BERT router with $\lambda = 0.7$, using DeBERTa-v3 as the BERT model, due to its superior performance in Section 5.2. Figure 3 presents the emissions and similarity scores for all routing methods, full results with confidence intervals can be found in Appendix D.8.

These results show our BERT router outperforms all the baseline methods, except the oracle router. This gap is expected; the oracle router is an upper bound for the performance of a router which is 100% accurate at predicting the best model.

The BERT router outperforms the most powerful single model (DeepSeek distilled Qwen-32B) by 3.3% whilst also reducing emissions by 39%. Our BERT router also outperforms generative (decoder) routing, whilst simultaneously decreasing the environmental cost. We also show the utilisation of the candidate LLMs by each router in Figure 4.

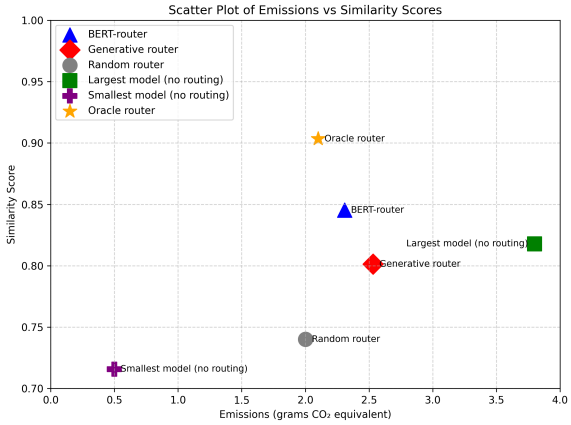

Figure 3: Similarity score vs average inference emissions, with $\lambda = 0.7$.

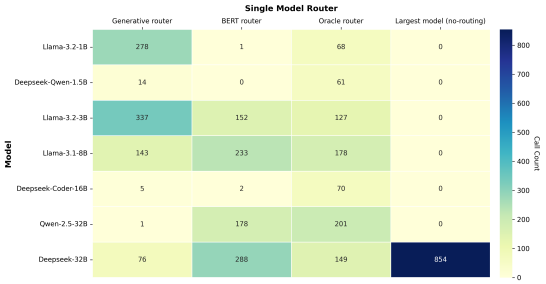

Figure 4: The number of calls to each model in single routing.

5.3.1 Unseen dataset Performance

In the previous section, we evaluated routing on a test set of 10% of the prompts from each benchmark dataset. We also reserve an entire benchmark dataset (SQUAD) from training, to test generalisability to new tasks (Rajpurkar et al., 2016). This addresses a limitation of similar research, where routing models were only tested on the benchmark datasets on which they were trained.

As shown in Figure 5, our router only reduces similarity scores by 0.51% compared to the largest model, whilst also reducing environmental costs by 12.5%. Full results with confidence intervals can be found in Appendix D.9.

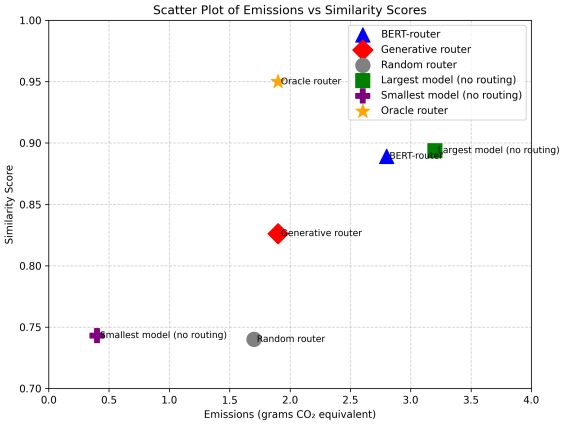

Figure 5: Similarity versus inference emission on unseen datasets, with $\lambda = 0.7$.

5.4 Chain Routing

5.5 Parameter Size Impact on Reasoning

We evaluate the impact of using larger models on each stage of reasoning CoTs in Figure 6.

These results show that smaller LLMs can effectively perform exploration and break down tasks, with under 2.4% impact on final similarity scores as opposed to larger models. However, they often fail to produce the correct final answer, even with the right reasoning chain, with the conclusion stage having by far the greatest impact on performance (14%). The average similarity score for each model combination is shown in Appendix D.7.

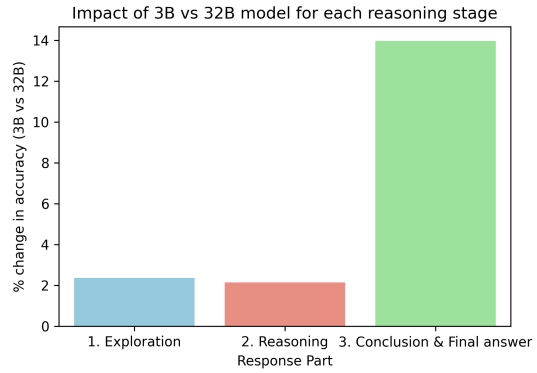

Figure 6: The percentage change in similarity scores when using the largest versus the smallest model for each part in the reasoning chain.

5.5.1 Chain Routing Inference Results

We evaluate the BERT router on the dataset of CoT responses in both modes: chain routing and single routing. To ensure fair results, we reduce the candidate LLMs available to the single model router to those used for chain routing and set $\lambda = 0.7$ in both cases. The results shown in Figure 7 highlight the potential of chain routing, which outperformed all methods except oracle routing. Full results with confidence intervals can be found in Appendix D.10. Notably, in comparison with the single largest model, our chain routing approach increases the similarity score by 0.43%, whilst decreasing the environmental cost by 18.7%.

Chain routing outperforms single routing on similarity scores by 4.84%, whilst only having a slightly higher environmental cost, highlighting the potential of chain routing as a more effective alternative to single routing. The headline figures for similarity score increase and emissions reduction are not as high as the previous single routing test, and single routing no longer outperforms the largest model. This can be explained by the fact that we use three models instead of seven, due to the exponentially increasing number of combinations in chain routing. We expect that scaling the number of models further would widen the margin and unlock further carbon savings.

The number of calls to each LLM is shown in Figure 8, showing that the BERT chain router mainly utilises the largest models. However, the chain router improves on the score of the largest model, showing that it is delegating to the smaller model at the correct times.

The number of calls to each model using the BERT router varies at different CoT stages. Smaller models are predominantly used for the first two stages (60 and 43 calls to the smallest model), and are used substantially less for the final step (17 calls to the smallest model). This mirrors the results found in Section 5.5, that output can be delegated to smaller models for the first two steps with a much lower impact on output quality than for the final step.

Breakdowns of the calls to each model for each CoT stage for both the generative and BERT routers are presented in Appendices D.11 and D.12.

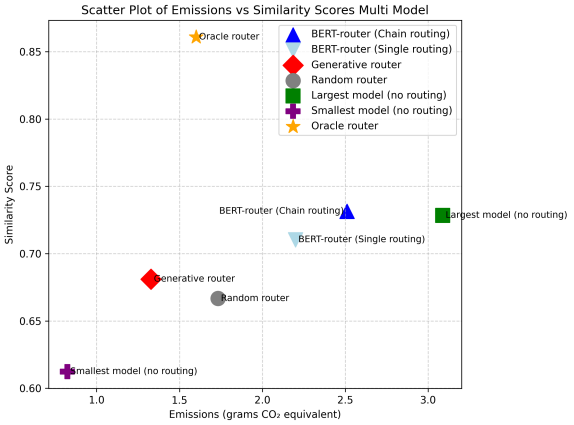

Figure 7: Similarity versus inference emission for routing on the CoT dataset, with $\lambda = 0.7$.

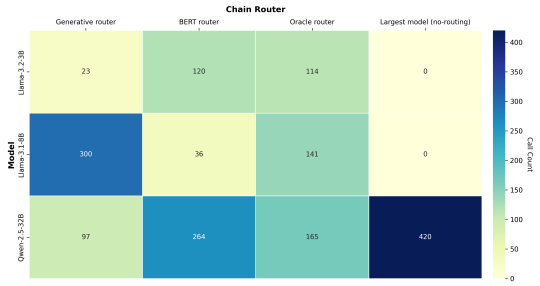

Figure 8: The number of calls to each model in chain routing.

5.6 Chain Routing Qualitative Error Analysis

We also manually analyse a sample of 50 prompts from the test set, to investigate the reliability of our similarity (to gold standard answer) metrics (BERTScore and chrF). Table 5 shows our comparison between the router's choice and the result with the highest metric. We record the number of times routing was successful/unsuccessful (whether the response from the model chosen by the router matched the gold standard response). We also record the number of cases where the similarity metric was correct/incorrect (whether the response with the highest similarity score matches the gold standard answer). Examples of CoT responses from the models chosen by the router and responses with the highest similarity score are shown in Appendix E.

6 Future Work

6.1 Additional Objectives

Whilst we focus on optimising output quality and environmental cost, the system could be extended to include other objectives. The leading candidate for this is the cost of inference; whilst this is difficult to quantify when running models locally, it is incorporated in related systems which rely on external APIs for inference (Hu et al., 2024).

Category	Count	%
Routing Quality		
Successful routing	42	84%
Unsuccessful routing	8	16%
Similarity Metric Quality		
Metric correct	46	92%
Metric incorrect	4	8%

Table 5: Qualitative error analysis of chain routing on a sample of 50 prompts.

6.2 Expanding Chain Routing

Our implementation of chain routing is a proof-of-concept to showcase the potential of this novel routing approach. This leaves several areas for future work to expand on, such as splitting the output into different stages and trying different model combinations, to further decrease environmental cost.

6.3 Improving Efficiency in Reasoning Models

Our results show that using smaller models for initial exploration steps in reasoning has little impact on final response quality, which could guide reasoning model development. Reasoning pipelines could incorporate a smaller model to explore the problem before passing the output to a larger model or MoE models could incorporate a dedicated lightweight expert for this phase.

7 Concluding Remarks

We incorporate environmental cost into routing decisions and introduce *chain routing*, which directs different stages of a reasoning CoT to different LLMs. We show that our system can significantly reduce the environmental impact of LLM inference, whilst also increasing the quality of outputs. Our results also show that using smaller models during early reasoning stages does not significantly affect final performance, a finding which can be used to create more efficient reasoning models. Overall, our findings highlight the substantial potential of environmentally aware chain routing in making LLM inference more sustainable.

Limitations

Generalisability and Dataset Selection

Whilst we attempt to make the system as generalisable as possible by including datasets spanning a wide range of domains, it is possible that the system may not generalise to all domains. The system may have to be fine-tuned further for deployment in tasks not included in the training data, such as dialogue.

Measuring similarity

Although research has shown that BERTScore and CHRF align well with human preference, they do not perfectly align (Zhang et al., 2020; Evtikhiev et al., 2023). However, we show from a manual qualitative review that the output with the highest similarity score matches the gold standard answer in 92% of cases.

Carbon Cost Estimation

The CodeCarbon library shows results broadly in line with results from previous research; models with more parameters or longer outputs consume more energy (Patterson et al., 2021). However, these results are based on batch inference on shared hardware (more details in Appendix C). Results in real-world deployment may differ dependent on deployment infrastructure, system load, and regional carbon intensity. However, we expect that all values will scale up or down in this case, and comparisons of the values between models will still hold.

Chain Routing Assumptions

Chain routing assumes that all reasoning processes can be segmented into the same three-phase structure; this aligns well with all tasks we use, and is likely to generalise to other tasks. However, it may not generalise to all tasks. Future work could explore chain routing with different methods for splitting CoT into stages.

Ethics Statement

Our work introduces a routing framework that considers both expected answer quality and environmental cost of inference in routing decisions. Whilst we hope that systems like ours will have a positive effect by decreasing the environmental cost of LLMs, it is important to address the potential ethical implications.

Carbon Cost of Training

The first consideration is the carbon cost and GPUs necessary to train the model. To address this, we carefully planned experiments to limit the amount of computing used to what was strictly necessary to train our system. We also only used a subset of prompts from each dataset and reduced the number of models in chain routing to ensure that we limit the environmental impact of gathering training data.

Transparency

Next, we need to consider transparency; systems like ours must make it clear at deployment time which model they are routing to, and allow for the end user to override this to ensure transparency and human agency.

Human Agency

The next consideration is the impact of these routing decisions; whilst we show that, on average, our routing increases performance, there will be some prompts where routing decreases the quality of responses. Routing could also have an impact on response hallucination rates. Thus, routing systems should always be deployed with a human-in-the-loop, to quality-check responses.

Bias

Finally, we need to consider bias, routing systems like ours are based on identifying statistical patterns in prompts to route them to the correct model. This could lead to prompts on certain topics or phrased in certain ways, being routed to smaller models, potentially reducing the quality of responses. At deployment time any system like ours would have to be carefully monitored and audited to ensure this was not happening.

References

Pranjal Aggarwal, Aman Madaan, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, Shyam Upadhyay, Manaal Faruqui, and Mausam. 2024. AutoMix: Automatically Mixing Language Models. In *Conference on Neural Information Processing System*.

Yadagiri Annepaka and Partha Pakray. 2025. Large language models: a survey of their development, capabilities, and applications. *Knowledge and Information Systems*, 67(3):2967–3022.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program Synthesis with Large Language Models. preprint.

Ljubisa Bojic, Predrag Kovacevic, and Milan Cabarkapa. 2023. GPT-4 Surpassing Human Performance in Linguistic Pragmatics. arXiv preprint. ArXiv:2312.09545 [cs].

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2025. A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*, 37(7):3896–3915.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. Frugalgpt: How to use large language models while reducing cost and improving performance. arXiv preprint.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. arXiv preprint. ArXiv:1803.05457 [cs].

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems. arXiv preprint. ArXiv:2110.14168 [cs].

Benoit Courty, Victor Schmidt, Sasha Luccioni, Goyal-Kamal, MarionCoutarel, Boris Feld, Jérémy Lecourt, LiamConnell, Amine Saboni, Inimaz, supatomic, Mathilde Léval, Luis Blanche, Alexis Cruveiller, ouminasara, Franklin Zhao, Aditya Joshi, Alexis Bogroff, Hugues de Lavoreille, Niko Laskaris, Edoardo Abati, Douglas Blank, Ziyao Wang, Armin Catovic, Marc Alencon, Michał Stęchły, Christian Bauer, Lucas Otávio N. de Araújo, JPW, and MinervaBooks. 2024. mlco2/codecarbon: v2.4.1.

Mehul Damani, Idan Shenfeld, Andi Peng, Andreea Bobu, and Jacob Andreas. 2024. Learning How Hard to Think: Input-Adaptive Allocation of LM Computation. In *International Conference on Learning Representations*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv preprint. ArXiv:2501.12948 [cs].

DeepSeek-AI, Qihao Zhu, Daya Guo, Zhihong Shao, Dejian Yang, Peiyi Wang, Runxin Xu, Y. Wu, Yukun Li, Huazuo Gao, Shirong Ma, Wangding Zeng, Xiao Bi, Zihui Gu, Hanwei Xu, Damai Dai, Kai Dong, Liyue Zhang, Yishi Piao, Zhibin Gou, Zhenda Xie, Zhewen Hao, Bingxuan Wang, Junxiao Song, Deli Chen, Xin Xie, Kang Guan, Yuxiang You, Aixin Liu, Qiushi Du, Wenjun Gao, Xuan Lu, Qinyu Chen, Yaohui Wang, Chengqi Deng, Jiashi Li, Chenggang Zhao, Chong Ruan, Fuli Luo, and Wenfeng Liang. 2024. DeepSeek-Coder-V2: Breaking the Barrier of Closed-Source Models in Code Intelligence. ArXiv:2406.11931 [cs].

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing. In *International Conference on Learning Representations*.

Ithier d'Aramon, Boris Ruf, and Marcin Detyniecki.

2024. Assessing Carbon Footprint Estimations of ChatGPT. In Philip Pong, editor, *Renewable Energy Resources and Conservation*, pages 127–133. Springer Nature Switzerland, Cham.

Mikhail Evtikhiev, Egor Bogomolov, Yaroslav Sokolov, and Timofey Bryksin. 2023. Out of the BLEU: how should we assess quality of the Code Generation models? *Journal of Systems and Software*, 203:111741.

Tao Feng, Yanzhen Shen, and Jiaxuan You. 2024. GraphRouter: A Graph-based Router for LLM Selections. In *International Conference on Learning Representations*.

Kevin Fischer, Darren Fürst, Sebastian Steindl, Jakob Lindner, and Ulrich Schäfer. 2024. Question: How do Large Language Models perform on the Question Answering tasks? Answer:. arXiv preprint.

Alon Goldstein, Miriam Havin, Roi Reichart, and Ariel Goldstein. 2023. Decoding Stumpers: Large Language Models vs. Human Problem-Solvers. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11644–11653. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun,

Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The Llama 3 Herd of Models. arXiv preprint.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025. Token-budget-aware llm reasoning. arXiv preprint.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing. In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*.

Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. 2024. RouterBench: A Benchmark for Multi-LLM Routing System. Poster presented at Agentic Markets Workshop, International Conference on Machine Learning.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive Mixtures of Local Experts. *Neural Computation*, 3(1):79–87. Conference Name: Neural Computation.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of Experts. arXiv preprint. ArXiv:2401.04088 [cs].

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics:*

EMNLP 2020, pages 1896–1907, Online. Association for Computational Linguistics.

Baolin Li, Yankai Jiang, Vijay Gadepally, and Devesh Tiwari. 2024. Sprout: Green Generative AI with Carbon-Efficient LLM Inference. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21799–21813. Association for Computational Linguistics.

Sasha Luccioni, Yacine Jernite, and Emma Strubell. 2024. Power Hungry Processing: Watts Driving the Cost of AI Deployment? In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '24, pages 85–99, New York, NY, USA. Association for Computing Machinery.

Meta. 2025. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation.

Seyed Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models. In *International Conference on Learning Representations*.

Brent Mittelstadt, Sandra Wachter, and Chris Russell. 2023. To protect science, we must use LLMs as zero-shot translators. *Nature Human Behaviour*, 7(11):1830–1832. Publisher: Nature Publishing Group.

Quang H. Nguyen, Duy C. Hoang, Juliette Decugis, Saurav Manchanda, Nitesh V. Chawla, and Khoa D. Doan. 2024. Metallm: A high-performant and cost-efficient dynamic framework for wrapping llms. arXiv preprint.

NVIDIA. 2025. NVIDIA-AI-Blueprints/llm-router.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Waleed Kadous, and Ion Stoica. 2024. RouteLLM: Learning to Route LLMs from Preference Data. In *International Conference on Learning Representations*.

OpenAI. 2024. Learning to Reason with LLMs.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon Emissions and Large Neural Network Training. arXiv preprint. ArXiv:2104.10350.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395. Association for Computational Linguistics.

Maja Popović. 2016. chrF deconstructed: beta parameters and n-gram weights. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 499–504, Berlin, Germany. Association for Computational Linguistics.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 Technical Report. arXiv preprint. ArXiv:2412.15115 [cs].

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Siddharth Samsi, Dan Zhao, Joseph McDonald, Baolin Li, Adam Michaleas, Michael Jones, William Bergeron, Jeremy Kepner, Devesh Tiwari, and Vijay Gadepally. 2023. From words to watts: Benchmarking the energy costs of large language model inference. In *2023 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–9.

Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. 2023. Large language model routing with benchmark datasets.

Toby Simonds, Kemal Kurniawan, and Jey Han Lau. 2024. MoDEM: Mixture of domain expert models. In *Proceedings of the 22nd Annual Workshop of the Australasian Language Technology Association*, pages 75–88, Canberra, Australia. Association for Computational Linguistics.

Kv Aditya Srivatsa, Kaushal Maurya, and Ekaterina Kochmar. 2024. Harnessing the power of multiple minds: Lessons learned from LLM routing. In *Proceedings of the Fifth Workshop on Insights from Negative Results in NLP*, pages 124–134, Mexico City, Mexico. Association for Computational Linguistics.

Dimitris Stripelis, Zhaozhuo Xu, Zijian Hu, Alay Dilipbhai Shah, Han Jin, Yuhang Yao, Jipeng Zhang, Tong Zhang, Salman Avestimehr, and Chaoyang He. 2024. TensorOpera router: A multi-model router for efficient LLM inference. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 452–462. Association for Computational Linguistics.

Clovis Varangot-Reille, Christophe Bouvard, Antoine Gourru, Mathieu Ciancone, Marion Schaeffer, and François Jacquenet. 2025. Doing More with Less – Implementing Routing Strategies in Large Language Model-Based Systems: An Extended Survey. arXiv preprint. ArXiv:2502.00409 [cs] version: 2.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024) Datasets and Benchmarks Track*.

Grant Wilkins, Srinivasan Keshav, and Richard Mortier. 2025. Offline Energy-Optimal LLM Serving: Workload-Based Energy Models for LLM Inference on Heterogeneous Systems. *SIGENERGY Energy Inform. Rev.*, 4(5):113–119.

Hao Xuan, Bokai Yang, and Xingyu Li. 2025. Exploring the impact of temperature scaling in softmax for classification and adversarial robustness.

Prateek Yadav, Colin Raffel, Mohammed Muqeeth, Lucas Caccia, Haokun Liu, Tianlong Chen, Mohit Bansal, Leshem Choshen, and Alessandro Sordoni. 2025. A survey on model moerging: Recycling and routing among specialized experts for collaborative learning. *Transactions on Machine Learning Research*. Survey Certification.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. 2023. Large Language Model Cascades with Mixture of Thought Representations for Cost-Efficient Reasoning. In *International Conference on Learning Representations*.

Kai Zhang, Liqian Peng, Congchao Wang, Alec Go, and Xiaozhong Liu. 2024a. LLM Cascade with Multi-Objective Optimal Consideration. ArXiv:2410.08014 version: 1.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations*.

Xuechen Zhang, Zijian Huang, Ege Onur Taga, Carlee Joe-Wong, Samet Oymak, and Jiasi Chen. 2024b. Efficient Contextual LLM Cascades through Budget-Constrained Policy Learning. In *International Conference on Learning Representations Poster*.

Wenhao Zheng, Yixiao Chen, Weitong Zhang, Souvik Kundu, Yun Li, Zhengzhong Liu, Eric P. Xing, Hongyi Wang, and Huaxiu Yao. 2024. CITER: Collaborative Inference for Efficient Large Language Model Decoding with Token-Level Routing. arXiv preprint.

Liu Zhuang, Lin Wayne, Shi Ya, and Zhao Jun. 2021. A robustly optimized BERT pre-training approach with post-training. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 1218–1227, Huhhot, China. Chinese Information Processing Society of China.

A Decoder Hyperparameters

A.1 LLM parameters ($< 25B$ parameters)

Setting	Value
Compute precision	`float16`
Top-k	50
Top-p	0.9
Temperature	0.7
Repetition penalty	1.2
Maximum output tokens	2500

Table 6: LLM hyperparameters for smaller models(<32B parameters).

A.2 LLM parameters ($> 25B$ parameters)

Setting	Value
Quantisation	8-bit
Compute precision	`float16`
Top-k	50
Top-p	0.9
Temperature	0.7
Repetition penalty	1.2
Maximum output tokens	2500

Table 7: LLM hyperparameters for larger models (>32B parameters).

B Encoder Hyperparameters

B.1 BERT Fine-Tuning Hyperparameters

Hyperparameter	Value
Learning rate	2×10^{-5}
Number of training epochs	3
Batch Size (Per GPU)	8
Warmup ratio	0.1
Weight decay	0.01
Mixed precision (fp16)	Enabled
Gradient accumulation steps	4
Soft score temperature	4.5

Table 8: BERT fine-tuning hyperparameters.

B.2 BERT Fine-Tuning Loss

The loss formula is shown in the equations below, where $P(X)$ is the correct soft score distribution, $Q(X)$ is the predicted soft score distribution, N is the total number of examples, k is the number of classes, and N_i is the number of examples in class i.

$$D_{KL}(P_i, Q_i) = \sum_j Q_{i,j} \log \frac{P_{i,j}}{Q_{i,j}} \tag{9}$$

$$w_i = \frac{N}{k \times N_i} \tag{10}$$

$$\text{loss}(Q_i, P_i, N_i) = D_{KL}(Q_i, P_i) \times w_i \tag{11}$$

C Carbon Calculation Details

C.1 Hardware Details

Component	Details
GPUs	• NVIDIA RTX A5000 : 23.99 GB • NVIDIA GeForce RTX 3090 : 24.00 GB
GPUS Utilised	At most 2 per experiment
System Memory	DDR4 3200MHz (250-500GB)
CPUs	• AMD EPYC 7543P : 32 cores, 2.8–3.7 GHz • AMD EPYC 7443 : 24 cores, 2.85–4.0 GHz
CPU Cores Utilised	At most 2 cores per experiment

Table 9: The hardware used for training and evaluation.

C.2 Code Carbon Implementation Details

We ran all of our results using the energy mix defined in Code Carbon for the United Kingdom (GBR), which uses an average grid-mix carbon intensity of 237.6g CO_2 per kWh [4].

These carbon intensity values are static, so they are the same between runs, but recreating our results in different jurisdictions or on different hardware will yield different results.

D Raw Results Output

D.1 Environmental Cost and Latency of Routing Approaches

Routing Method	Carbon Cost (g CO_2)	Latency (s)
BERT (Single Routing)	0.0001565	0.0251
BERT (Chain Routing)	0.04102	0.8969
Generative (Single Routing)	0.01565	1.004
Random, Oracle Routing	0	0

Table 10: A comparison of the environmental cost of each routing approach.

D.2 Environmental Cost of Training BERT Routers

Router	Carbon cost (total) CO_2 kg	Carbon cost (per training example) CO_2 kg
Bert (single routing)	0.0364	0.00000474
BERT (chain routing)	0.809	0.00000927

Table 11: The carbon cost of training the BERT router for both single and chain routing

[4] https://github.com/mlco2/codecarbon/blob/master/codecarbon/data/private_infra/global_energy_mix.json

D.3 Environmental Cost of Gathering Training Dataset

Model	Total Cost (kg CO_2)
deepseek-ai/DeepSeek-R1-Distill-Qwen-32B	35.9
Qwen/Qwen2.5-32B	26.5
deepseek-ai/DeepSeek-Coder-V2-Lite-Instruct	21.2
deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B	16.3
meta-llama/Llama-3.1-8B	10.8
meta-llama/Llama-3.2-3B	8.3
meta-llama/Llama-3.2-1B	4.9
TOTAL	**123.9**

Table 12: Carbon costs by model for single routing.

Models	Total Cost (kg CO_2)
32B-32B-32B	4.10
3B-32B-3B	3.81
32B-32B-8B	3.11
32B-32B-3B	3.05
3B-32B-8B	2.76
3B-32B-32B	2.73
8B-32B-3B	2.19
8B-32B-8B	2.17
8B-32B-32B	2.16
3B-8B-32B	2.16
3B-3B-32B	2.08
32B-3B-8B	2.00
32B-8B-3B	1.97
8B-8B-3B	1.95
32B-8B-32B	1.91
8B-3B-3B	1.89
32B-3B-32B	1.66
8B-8B-8B	1.65
32B-3B-3B	1.63
8B-3B-8B	1.59
3B-8B-3B	1.58
8B-8B-32B	1.50
8B-3B-32B	1.39
3B-8B-8B	1.30
3B-3B-8B	1.69
3B-3B-3B	1.07
TOTAL	**56.9**

Table 13: Carbon costs by model combination for chain routing.

D.4 Lambda Trade-off Results

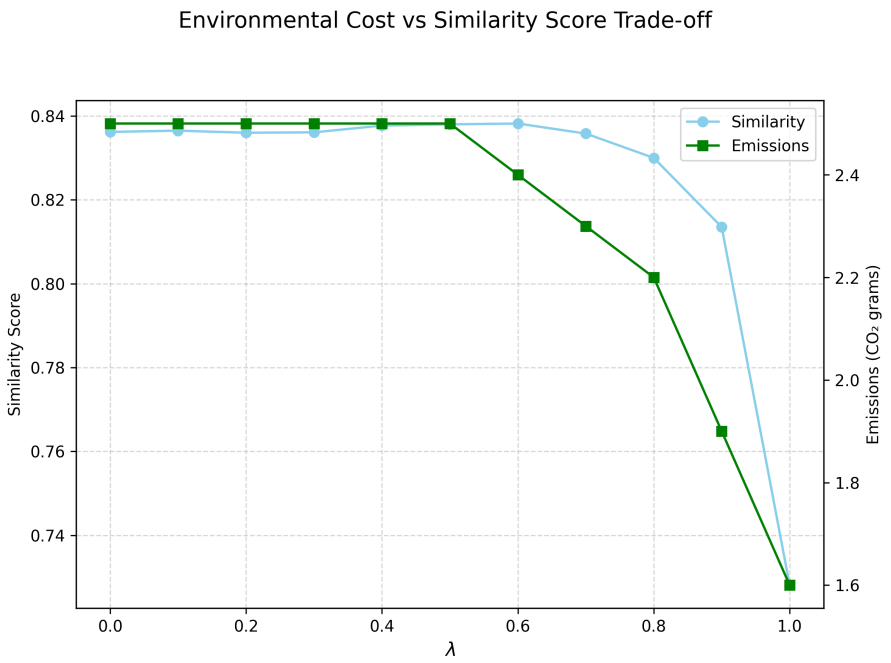

Figure 9: A graph showing the similarity score and emissions for different λ values.

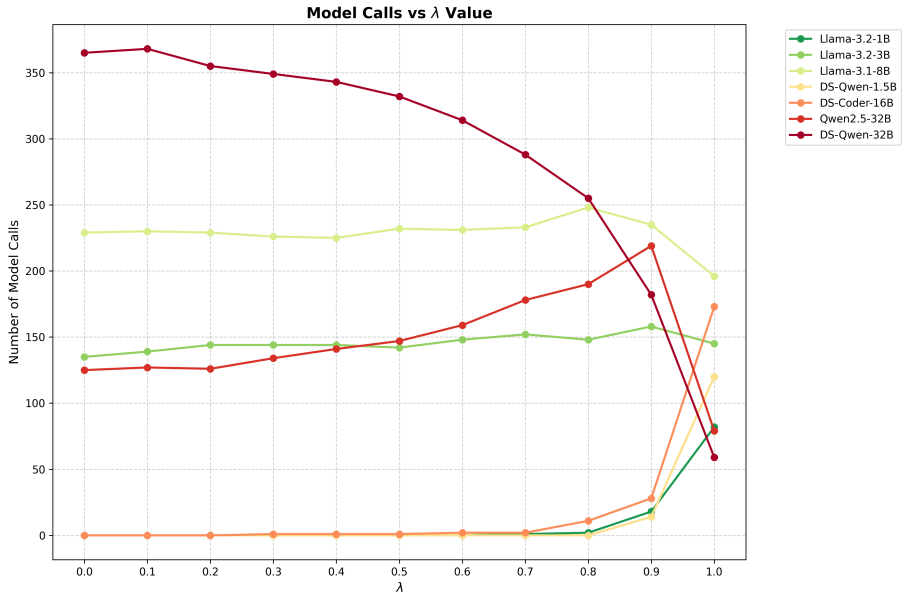

Figure 10: The number of calls to each candidate model against the value of λ.

Model/ λ	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Llama-3.2-1B	0	0	0	0	0	0	0	1	2	18	82
Llama-3.2-3B	135	139	144	144	144	142	148	152	148	158	145
Llama-3.1-8B	229	230	229	226	225	232	231	233	248	235	196
DS-Qwen-1.5B	0	0	0	0	0	0	0	0	0	14	120
DS-Coder-16B	0	0	0	1	1	1	2	2	11	28	173
Qwen2.5-32B	125	127	126	134	141	147	159	178	190	219	79
DS-Qwen-32B	365	368	355	349	343	332	314	288	255	182	59

Table 14: A table showing the number of calls to each candidate model for different λ values in single routing.

D.5 Similarity Scores by Model and Task for Single Routing

Task	Qwen2.5-32B	DS-Coder-16B	DS-Qwen-1.5B	DS-Qwen-32B	Llama-3.1-8B	Llama-3.2-3B	Llama-3.2-1B
GSM-Symbolic	0.787	0.744	0.218	0.735	0.964	0.962	0.773
SQuAD	0.877	0.737	0.668	0.894	0.813	0.855	0.743
ARC	0.791	0.822	0.726	0.877	0.835	0.838	0.686
MBPP	0.179	0.194	0.090	0.142	0.238	0.116	0.189
MMLU	0.850	0.723	0.745	0.857	0.792	0.839	0.704
MMLU-Pro	0.902	0.745	0.754	0.890	0.807	0.863	0.805
GSM8K	0.971	0.773	0.681	0.979	0.987	0.959	0.803
GPQA	0.794	0.702	0.709	0.808	0.818	0.824	0.725
UnifiedQA	0.752	0.650	0.665	0.784	0.771	0.769	0.685

Table 15: Average similarity scores by task and model.

D.6 Similarity Scores by Model and Task for Chain Routing

Model	GSM-Symbolic	ARC	MBPP	MMLU	MMLU-Pro	GSM8K	UnifiedQA
32B-32B-32B	0.8990	0.7792	0.2089	0.8669	0.8409	0.8629	0.7122
32B-32B-8B	0.6214	0.8075	0.2281	0.8540	0.8075	0.6674	0.7449
32B-32B-3B	0.5959	0.7473	0.2140	0.7982	0.7931	0.6059	0.6766
32B-8B-32B	0.8445	0.7911	0.1777	0.8974	0.8462	0.7808	0.7161
32B-8B-8B	0.7020	0.7428	0.1899	0.8355	0.7837	0.7163	0.6793
32B-8B-3B	0.6045	0.7215	0.1782	0.8120	0.7930	0.5939	0.6451
32B-3B-32B	0.7380	0.7902	0.1814	0.8809	0.8243	0.7235	0.7147
32B-3B-8B	0.6651	0.7572	0.2097	0.8390	0.8073	0.7177	0.7147
32B-3B-3B	0.5846	0.7409	0.1988	0.8210	0.7914	0.6222	0.6606
8B-32B-32B	0.7883	0.7884	0.1732	0.9158	0.8763	0.7877	0.7135
8B-32B-8B	0.7171	0.7844	0.2012	0.8522	0.7971	0.7381	0.7194
8B-32B-3B	0.6223	0.7192	0.2056	0.8060	0.7994	0.5967	0.6682
8B-8B-32B	0.8009	0.8071	0.1565	0.9193	0.8587	0.7779	0.7325
8B-8B-8B	0.8279	0.7602	0.1831	0.8342	0.8032	0.7950	0.7026
8B-8B-3B	0.6304	0.7081	0.1705	0.7930	0.7942	0.6311	0.6436
8B-3B-32B	0.7309	0.7984	0.1753	0.8769	0.8522	0.6958	0.7329
8B-3B-8B	0.7219	0.7808	0.2064	0.8347	0.8150	0.6720	0.7279
8B-3B-3B	0.6318	0.6874	0.1925	0.8124	0.7762	0.6195	0.6345
3B-32B-32B	0.7300	0.7870	0.1753	0.9020	0.8366	0.6989	0.7200
3B-32B-8B	0.7324	0.7559	0.2188	0.8451	0.7874	0.6983	0.6876
3B-32B-3B	0.6208	0.7044	0.2037	0.7959	0.7967	0.5472	0.6310
3B-8B-32B	0.7507	0.8034	0.1505	0.9125	0.8328	0.7372	0.7100
3B-8B-8B	0.7293	0.7464	0.1807	0.8216	0.7667	0.7519	0.6662
3B-8B-3B	0.6103	0.6927	0.1823	0.8095	0.8060	0.5851	0.6390
3B-3B-32B	0.7051	0.7919	0.1715	0.8846	0.8493	0.6578	0.7038
3B-3B-8B	0.7087	0.7596	0.2043	0.8497	0.8058	0.7213	0.6933
3B-3B-3B	0.5946	0.7168	0.1932	0.7923	0.8002	0.5514	0.6282

Table 16: Average similarity scores by model and task.

D.7 Chain Routing Model Performance

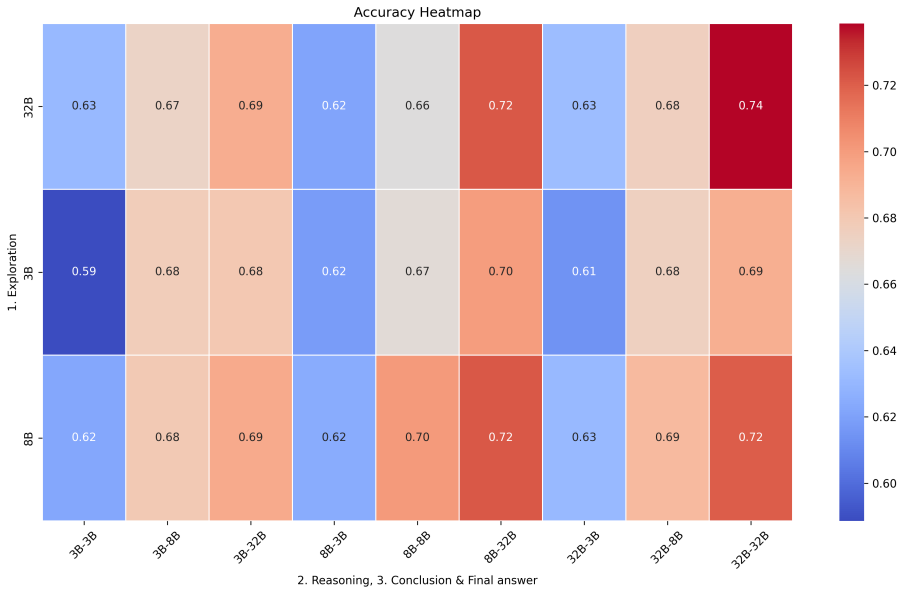

Figure 11: A heatmap showing average performance across datasets for every combination of models

D.8 Single Routing Similarity and Emissions Results with Confidence Intervals

Router	Emissions (grams CO_2)	Similarity Score
BERT-router	2.31 ± 0.004	0.845 ± 0.0264
Generative router	2.53 ± 0.002	0.81 ± 0.0326
Random router	2.00 ± 0.008	0.74 ± 0.101
Largest model (no routing)	3.80 ± 0.00600	0.818 ± 0.0302
Smallest model (no routing)	0.500 ± 0.002	0.716 ± 0.0330
Oracle router	2.10 ± 0.032	0.903 ± 0.644

Table 17: Model emissions and similarity scores with 95% confidence intervals for single routing.

D.9 Single Routing Similarity and Emissions Results on Unseen Dataset with Confidence Intervals

Router	Emissions (grams CO_2)	Similarity Score
BERT-router	2.80 ± 0.002	0.889 ± 0.0131
Generative router	1.90 ± 0.001	0.826 ± 0.0155
Random router	1.70 ± 0.002	0.74 ± 0.0176
Largest model (no routing)	3.20 ± 0.001	0.894 ± 0.0135
Smallest model (no routing)	0.40 ± 0.002	0.743 ± 0.0188
Oracle router	1.90 ± 0.002	0.950 ± 0.0057

Table 18: Model emissions and similarity scores with 95% confidence intervals for single routing on unseen dataset.

D.10 Chain Routing Similarity and Emissions Results with Confidence Intervals

Router	Emissions (grams CO_2)	Similarity Score
BERT-router (Chain)	2.51 ± 0.00441	0.732 ± 0.0479
BERT-router (Single)	2.20 ± 0.002	0.710 ± 0.0231
Generative router	1.33 ± 0.00176	0.681 ± 0.1282
Random router	1.73 ± 0.002	0.667 ± 0.0176
Largest model (no routing)	3.09 ± 0.005	0.728 ± 0.04787
Smallest model (no routing)	0.824 ± 0.006	0.612 ± 0.0463
Oracle router	1.90 ± 0.00212	0.861 ± 0.01365

Table 19: Model emissions and similarity scores with 95% confidence intervals for chain routing.

D.11 Chain Routing model usage per stage for BERT Router

Stage	LLama 3.2-3B	Llama 3.1-8B	Qwen 2.5-32B
1 (Exploration)	60	10	70
2 (Reasoning)	43	9	88
3 (Reasoning)	17	17	106

Table 20: The number of calls for each model for each CoT stage in the chain router when using the BERT router.

D.12 Chain Routing Model Usage per Stage for Generative Router

Stage	LLama 3.2-3B	Llama 3.1-8B	Qwen 2.5-32B
1 (Exploration)	33	91	16
2 (Reasoning)	30	100	10
3 (Reasoning)	24	108	8

Table 21: The number of calls for each model for each CoT stage in the chain router when using the generative router.

D.13 Example Prompt

System prompt
System: You are a helpful AI assistant. You provide clear, short, step-by-step solutions and always conclude with #### followed by the final answer. **User:** Now, answer the following question: Which of the following was not cited as evidence for life in the Martian meteorite ALH84001? **<Options:>** Please provide a short, step-by-step solution to the question above, and conclude with the #### followed by the final answer.

Table 22: Example system prompt.

D.14 System Prompt for Reasoning Chain Decomposition

System prompt
System: Split your response into three distinct parts: initial EXPLORATION, REASONING, and a CONCLUSION containing #### followed by your final answer for evaluation. Format your response with the following structure: EXPLORATION: your initial thoughts REASONING: your step-by-step reasoning CONCLUSION:#### your final answer

Table 23: Example system prompt for reasoning chain decomposition.

D.15 Example CoT Response Split Into Stages

Stage	Output
Prompt	A school board of a large school district is proposing a new dress code for students. Some students feel that this dress code unfairly targets female students. To see if there is a difference between boys and girls in their opposition to the new dress code, they conduct a poll of 60 randomly selected male and 70 randomly selected female high school students in the district. They find that 66 females oppose the dress code and 50 males oppose the dress code. Which of the following explains why a two-proportion z-test is not appropriate?
Exploration	The question is asking why a two-proportion z-test is not appropriate for the given scenario. A two-proportion z-test is used to compare the proportions of two independent groups. In this case, the groups are male and female students who oppose the new dress code.
Reasoning	The sample sizes are 60 males and 70 females, and the number of successes (opposition) are 50 males and 66 females. To determine if a two-proportion z-test is appropriate, we need to check the conditions for the test. The conditions are: 1. The samples are independent. 2. The sample sizes are large enough, which means that the number of successes and failures in each group should be at least 10. In this case, the samples are independent because they are randomly selected male and female students. However, the number of successes and failures in each group should be checked:- For males: 50 successes (opposition) and 10 failures (not opposition) For females: 66 successes (opposition) and 4 failures (not opposition)The number of failures for the female group is less than 10, which violates the condition for a two-proportion z-test.
Conclusion	Therefore, a two-proportion z-test is not appropriate for this scenario. #### The number of successes and the number of failures for the two groups are not all large enough.

Table 24: Example chain routing output (from Qwen-32B on MMLU dataset).

E Example Responses From Qualitative Analysis

E.1 Example Correct CoT Response

Stage	Output
Question	If the money supply increases what happens in the money market (assuming money demand is downward sloping)?
Gold Standard	The nominal interest rates falls
Router Output	CONCLUSION: The nominal interest rates falls

Table 25: An example of where the CoT response from the router was deemed to be correct from qualitative review.

E.2 Example Incorrect CoT Response

Stage	Output
Prompt	A farmer is buying feed for his horses. He buys a variety of oats, hay, carrots and apples. Since apples are a rare treat, he only buys four 3-pound boxes of them for the whole stable. He only wants enough carrots to feed the horses while the vegetables are fresh, so he buys thirteen 9-pound bags. Oats are the main diet of his horses, so he buys thirty four 44-pound packs. Hay are a staple to supplement the oats, so he buys fifteen 19-pound sacks. Finally, he buys 43 pounds of grocery, and 27 pounds of tools. A farm truck can carry 220 pounds at a time. How many trips does the farmer need to transport all the items, if he has three trucks?
Gold Standard	3
Output	4

Table 26: An example of where the CoT response from the router was deemed to be incorrect from qualitative review.

E.3 Example Correct Highest Similarity Score Response

Stage	Output
Prompt	Let l = [1,2,3,4]. What is max(l) in Python3?
Gold Standard	4
Output	4
Similarity	1.00

Table 27: An example of where the response with the highest similarity score matched the gold standard output.

E.4 Example Incorrect Highest Similarity Score Response

Stage	Output
Prompt	Let a undirected graph G with edges E = <2,1>,<2,0>,<2,3>,<1,4>,<4,3>, which <A,B> represent Node A is connected to Node B. What is the minimum vertex cover of G? Represent the vertex cover in a list of ascending order.
Gold Standard	[2, 4]
Output	The final answer is boxed0 2 3 4
Similarity	0.8235

Table 28: An example of where the response with the highest similarity score did not match the gold standard output.