COMPSCI 590.01 FINAL PROJECT REPORT

Zhuofan Jia, Erick Jiang, Berna Kötehne, Tianqing Li & Xinyuan Lyu
Duke University
{zj88,ej127,bk214,tl299,xl481}@duke.edu

ABSTRACT

This study investigates optimization strategies for the Gemma-3-270M model across various tasks: factual question answering, multi-step reasoning, and instruction following. We train a router to predict task types and dispatch inputs to specialized expert models. For factual QA, we implement BM25-based retrieval augmentation. For reasoning, we apply knowledge distillation from a 1B-parameter teacher model using both soft logit targets and chain-of-thought supervision to transfer reasoning capabilities. For instruction following, we perform supervised fine-tuning on diverse instruction datasets. Our integrated system achieves 20.72% exact match (relaxed) on TriviaQA (2× baseline), 26.63% accuracy on ARC-Challenge, and 39.23% loose instruction accuracy on IFEval, exceeding target performance across all benchmarks.

1 INTRODUCTION

Large language models (LLMs) exhibit strong emergent abilities such as factual knowledge recall, multi-step reasoning, and instruction following (Wei et al., 2022a; Ouyang et al., 2022), but these capabilities typically rely on models with billions of parameters (Kaplan et al., 2020; Hoffmann et al., 2022). While highly effective, such models are expensive to train, fine-tune, and deploy, motivating growing interest in small language models (SLMs) that offer improved efficiency and accessibility. However, due to severe capacity constraints, SLMs often struggle with knowledge retention, reliable reasoning, and generalization across tasks (Xu et al., 2024), leaving open the question of how far their performance can be pushed with modern training techniques.

Recent work shows that compact models can achieve competitive performance when trained with carefully curated data and effective post-training strategies. Models such as Phi-2 (Javaheripi & Bubeck, 2023), TinyLlama (Zhang et al., 2024), and the Gemma family (Team et al., 2025) highlight the role of data quality, synthetic data generation, and distillation in improving efficiency–quality trade-offs. In addition, instruction tuning and diverse data mixtures have been shown to significantly improve generalization in smaller models (Chung et al., 2022; Chen et al., 2023). Building on these insights, we push this line of inquiry to an extreme setting by studying a 270M-parameter model, far smaller than the scale at which LLM abilities typically emerge, and investigate how different adaptation mechanisms interact under such stringent capacity limitations.

In this project, we evaluate the Gemma-3 270M model across three complementary task categories: factual question answering, reasoning, and instruction following. Rather than relying on a single fine-tuning recipe, we adopt a modular approach that combines supervised fine-tuning (SFT) (Ouyang et al., 2022), knowledge distillation (KD) (Hinton et al., 2015; Hsieh et al., 2023), retrieval-augmented generation (RAG) (Lewis et al., 2020), and a learned routing mechanism that assigns inputs to task-specialized experts (Shazeer et al., 2017). Our results show that this integrated approach substantially outperforms zero-shot and single-method baselines across all three benchmarks. Notably, combining distillation, retrieval, and routing enables the 270M model to exceed target performance levels previously thought achievable only by significantly larger models, demonstrating that careful system design can meaningfully extend the capabilities of SLMs.

2 Related Work

Instruction tuning and supervised fine-tuning (SFT) are standard approaches for adapting pretrained language models. InstructGPT (Ouyang et al., 2022) and FLAN (Chung et al., 2022) show that diverse instruction datasets improve generalization across tasks, while synthetic data generation further expands task coverage and robustness (Chen et al., 2023). Parameter-efficient methods such as LoRA (Hu et al., 2021) and QLoRA (Dettmers et al., 2023) enable adaptation under strict memory constraints by updating only a small subset of parameters. Our work focuses on *full-parameter* SFT for a 270M-parameter model and systematically probes how different data mixtures affect training stability and catastrophic forgetting, providing a controlled view of what can be achieved without parameter-efficient tricks.

Knowledge distillation (KD) transfers capabilities from large teacher models to smaller students (Hinton et al., 2015; Tang et al., 2020; Xu et al., 2024). Beyond standard label or logits matching, chain-of-thought (CoT) distillation shows that exposing students to intermediate reasoning steps can substantially improve reasoning performance (Wei et al., 2022b; Hsieh et al., 2023). We build on this line of work and run dislation from a Gemma3-1B-IT teacher and analyze different strategies in our severely capacity-limited 270M model.

Retrieval-augmented generation (RAG) mitigates the limited parametric knowledge of language models by incorporating external memory, improving factual accuracy and reducing hallucinations (Lewis et al., 2020; Asai et al., 2023). This is particularly attractive for SLMs, whose internal knowledge capacity is inherently limited. In parallel, routing and mixture-of-experts (MoE) methods (Shazeer et al., 2017) increase effective capacity by dynamically selecting specialized components, but are often studied at much larger scales. In our work, we combine these ideas in a lightweight way: we use RAG to complement the factual deficits of a 270M model and adopt a simple task-level routing mechanism in which a classifier assigns inputs to fine-tuned specialists, forming a coarse-grained MoE design tailored to small models.

3 Approach

3.1 Background

We consider a sequence-to-sequence setting where a LM receives an input prompt $x = (x_1, \ldots, x_m)$ and generates an output sequence $y = (y_1, \ldots, y_n)$ by $p_\theta(y \mid x) = \prod_{i=1}^{n} p_\theta(y_i \mid x, y_{1:i-1})$ with θ denoting model parameters. Our system is built around a small Gemma-3 270M model specialized into multiple experts for three broad tasks, each of which requires distinct behaviors, yet task labels are not available at test time, and is orchestrated by a learned router $y = \mathrm{Expert}_{r(x)}(x)$. All experts share the same initialization but are adapted by different techniques as we elaborate later. This modular design enables us to study how adaptation strategies interact under strict capacity constraints and whether a collection of small specialists can outperform a single fine-tuned model.

3.2 Evaluation

We evaluate our system on three public benchmarks: **TriviaQA** (Joshi et al., 2017) (factual question answering) challenges the model with complex, open-domain questions and we report the relaxed Exact Match. **ARC-Challenge** (Clark et al., 2018) (scientific reasoning) contains grade-school level, 4-way multiple-choice question. We report accuracy. **IFEval** (Zhou et al., 2023) (instruction following) evaluates whether the model follows explicit prompt constraints. We report loose instruction accuracy. All datasets are evaluated in a unified prediction format: the router assigns each input to an expert model, the selected expert generates a response, and predictions are compared against task-specific gold annotations. We additionally report performance on the hidden evaluation set provided by course staff, which is used only for final scoring and not for model development.

3.3 Training an LLM-based router

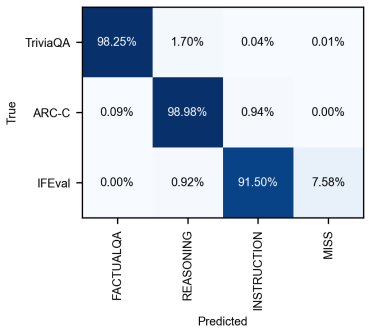
As the underlying benchmark labels cannot be leveraged for fairness, we trained a separate Gemma3-270M router that infers task type by *generating* the text label (e.g., "factual QA", "reasoning", "instruction following"). Figure 1a illustrates how the training samples are composed.

```
'input': 'Can you provide a list of 3 reasons why the Fachschule für
Luftfahrzeugführer (FFL) is considered an excellent institution for
aviation training, ensuring that each point includes the keywords:
"aviation", "training", and "Germany"? Additionally, analyze the
frequency of the letter \'a\' in the entire response.'
'output': 'INSTRUCTION'

'input': 'Popular mainly since the 1960s, what ten letter word is the
name of the lightweight waist-length jacket made of cotton, polyester,
wool or suede, usually with traditional Fraser tartan or check
patterned lining? It is said to have earned its nickname due to the
fact that it was often worn by one of the lead characters of the TV
show Peyton Place.'
'output': 'FACTUALQA'

'input': 'Some meteorologists use visual aids such as diagrams, maps,
and charts when they present the weather to television viewers. How do
different types of visual aids help the viewers understand complex
weather phenomena.'
'output': 'REASONING'
```

(a) Example training samples.



(b) Classification performance on the benchmarks.

Figure 1: Router LM.

At inference, the router's prediction is mapped to one of the three expert models from searching the first occurrence of task types. The router achieves a macro-averaged accuracy of 96.24% (Figure 1b, used datasets found in Table 3), ensuring robust routing.

3.4 Supervised Fine-Tuning (SFT)

SFT adapts a pretrained base model with general competence to task- or domain-specific abilities. The training objective uses the standard autoregressive next-token cross-entropy loss for LLMs, which minimizes the negative log-likelihood of future tokens given the past: $\mathcal{L}_{\text{SFT}} = -\sum_{i=1}^{N} \log p_\theta(y_i \mid y_{1:i-1})$ where θ denotes the model parameters. Raw instruction–response pairs are formatted using a chat template into a single text sequence, with different roles (e.g., `<system>`, `<user>`, `<assistant>`) clearly separated by special tokens. In our setting, the main concern is the *quality* and *distribution* of the training corpus versus absolute quantity. The limited capacity of a 270M-parameter model makes it particularly prone to catastrophic forgetting: after fine-tuning on a narrow task (e.g., multiple-choice QA), the model may overfit and lose its ability to generate coherent responses or generalize to new tasks. While highly specialized experts are acceptable—since the router directs inputs to the most suitable expert—it is still important to understand the SLM's behavior under SFT and to strike a balance between specialization and generality, given the limited number of parameters available to store information.

3.5 Knowledge Distillation (KD)

Knowledge distillation is a model compression technique where a compact *Student* model is trained to replicate the behavior and performance of a larger, more capable *Teacher*. In our setting, KD aims not only to transfer knowledge, but also to convey the Teacher's reasoning patterns, enabling the small model with better generalization than with raw SFT alone (Hinton et al., 2015; Tang et al., 2020; Xu et al., 2024). We use Gemma3-1B-IT as the Teacher and evaluate two major KD strategies:

Logits-matching KD: Beyond matching final outputs, the Student is trained to approximate the Teacher's full logit distribution over the vocabulary. The objective combines standard cross-entropy with a KL divergence term:

$$\mathcal{L} = (1-\alpha)\,\mathcal{L}_{CE} + \alpha\,\mathcal{L}_{KD} = (1-\alpha)\left[-\sum_{i=1}^{V} y_i \log q_i(1)\right] + \alpha\left[-\tau^2 \sum_{i=1}^{V} p_i(\tau) \log q_i(\tau)\right] \quad (1)$$

where $p_i(\tau)$ and $q_i(\tau)$ are teacher and student distributions at temperature τ, y_i is the one-hot ground-truth label, and V is the vocabulary size. Increasing τ smooths the distribution, exposing relative token likelihoods and encouraging the Student to learn structural similarities than overfitting to sharp predictions. The τ^2 factor compensates for softmax scaling and prevents gradient vanishing.

User

Question: Lichens are symbiotic organisms made of green algae and fungi. What do the green algae supply to the fungi in this symbiotic relationship? The correct Answer is B) Food. Provide a step-by-step reasoning process that leads to this correct answer.

Assistant Response

Here is a step-by-step reasoning process to arrive at the correct answer:

1. **Understanding the symbiotic relationship:** Lichens are a classic example of a *mutualistic relationship*—both organisms benefit from the interaction. The fungi provide the structure and protection, and the algae provide the food.
2. **Analyzing the options: A) Carbon dioxide:** While algae *do* produce carbon dioxide... **B) Food: This is the most accurate description.** The algae provide the fungi with carbohydrates (sugars) produced through photosynthesis [IGNORE ...]
3. **Conclusion:** Therefore, the green algae supply food to the fungi.

Figure 2: Example of Teacher model response using chain-of-thought prompts.

Chain-of-thought (CoT) KD: Small models often lack the ability to generate multi-step reasoning, even when instruction-tuned. To address this, we prompt the Teacher to produce detailed, step-by-step reasoning traces, which are then used as supervision targets alongside standard training data. This provides the Student with explicit intermediate reasoning signals. An example CoT annotation from the Teacher is shown in Figure 2.

3.6 Retrieval-Augmented Generation (RAG)

Gemma3-270M lacks the parametric capacity to store broad world knowledge, making closed-book factual QA particularly difficult. To compensate, we attach a lightweight RAG pipeline that supplies non-parametric memory at inference time.

We construct two BM25-based knowledge bases (Robertson & Zaragoza, 2009). The first uses MS MARCO passages (Bajaj et al., 2016) (8.8M documents; avg. 242 tokens, median 233). The second uses Simple English Wikipedia (Wikimedia Foundation) (241K documents; avg. 1124 tokens, median 890). Although SimpleWiki retrieves the gold answer more frequently (32.1% vs. 28.7%), its longer contexts degrade QA accuracy, as the SLM struggles to extract relevant evidence from lengthy inputs. We therefore adopt MS MARCO as our default knowledge base.

As shown in Figure 3, for each question we retrieve the top-k passages offline (with $k = 3$ in our implementation) and concatenate them with the question as model input. The model receives prompts instructing it to use the contextual evidence, falling back to closed-book QA when no evidence is available. Additional details are presented in Appendix A.3.

Figure 3: Overview of the retrieval-augmented generation (RAG) pipeline.

We also experimented with a simplified Self-RAG-style re-ranking approach (Asai et al., 2023) using a learned evaluator. Given the top-k retrieved contexts, we generate k multiple candidate answers and score them with a separately fine-tuned Gemma3-270M evaluator, which assigns scores via a single forward pass based on logits corresponding to relevance, factual support, and utility. The highest-scoring candidate is then selected (see Appendix A.4 for details). In practice, this re-ranking strategy did not outperform the standard RAG baseline: the evaluator struggled to reliably distinguish subtle errors, and generating multiple candidates introduced additional variance. Consequently, we use the BM25-based RAG pipeline as our final retrieval method.

4 Experiments and Analysis

4.1 Results for Supervised Fine-Tuning (SFT)

We evaluate several SFT strategies designed to isolate how training data composition influences a small model's stability and downstream performance: (1) **QA-only**: QA pairs from ARC-Challenge, ARC-Easy, and TriviaQA; (2) **IF-only**: instruction-following data from TULU-3; (3) **Data-Mixture**: diverse mixtures combining instruction-following, conversation, math, and coding datasets (Table 4). Data-Mixture-1 includes TULU-3, SmolTalk, and OpenMathInstruct and Data-Mixture-2 further incorporates AutoIF and ORCA samples.

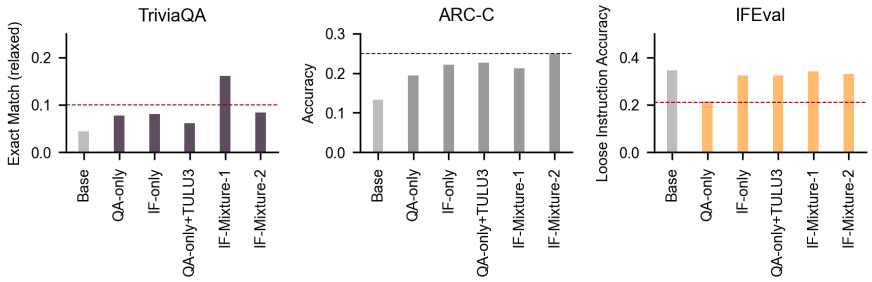


Figure 4: Performance of SFT models trained on different data mixtures.

As shown in Figure 4, the choice of training corpus strongly affects downstream performance and stability. Training only on QA pairs causes catastrophic forgetting of general knowledge, resulting in degraded instruction following and incoherence. The IF-only model performs better on instruction adherence and reasoning. The mixed-data strategies (especially Mixture-1) provide the most balanced results, reducing forgetting while improving performance across all three benchmarks.

4.2 Results for Knowledge Distillation (KD)

We compared the performance of logits-matching KD, CoT KD, and a hybrid method that combines both (Figure 5). While the zero-shot base model shows limited ability on ARC-Challenge (about 13.3%), both distillation approaches produce substantial improvements. Logits-matching KD increases accuracy to roughly 21%, and adding explicit reasoning traces through CoT KD further raises performance to around 24%. The hybrid method (KD-both) yields the best results, indicating that soft probability targets and step-by-step reasoning provide complementary benefits for reasoning-heavy tasks. As expected, these gains come with a reduction in instruction-following performance compared to the base model, so we did not use KD for this task in our final model.

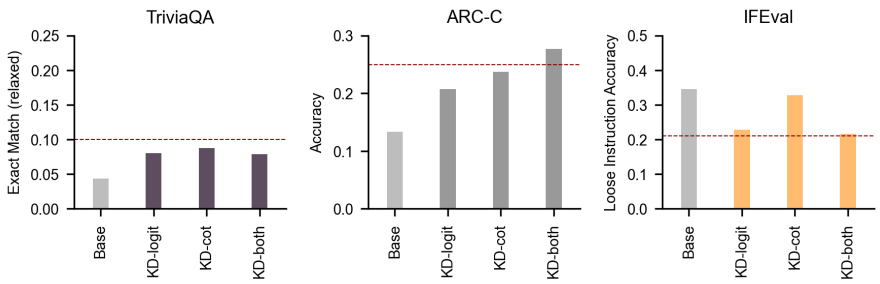


Figure 5: Comparison of KD strategies. Combining logits-based and CoT-based signals yields the best reasoning performance, outperforming either method alone.

4.3 Results for RAG

We now evaluate the benefit of retrieval on factual QA. Table 1 reports TriviaQA performance under different corpus choices and re-ranking strategies. Using the MS MARCO index yields the best improvement (EM = 0.2072). SimpleWiki retrieves more gold-containing passages but harms accuracy due to its significantly longer context lengths, which overwhelm the 270M model. A simplified Self-RAG-style re-ranking variant (Asai et al., 2023) reduces accuracy, as the evaluator struggles to differentiate subtle factual errors and multi-candidate generation increases variance.

Table 1: TriviaQA performance using RAG. All RAG settings use router to dispatch factual QA queries to Gemma3-270M with retrieved context.

Setting	**TriviaQA EM**
Base model (no router, no RAG)	0.044
RAG + SimpleWiki	0.1694
RAG + MS MARCO (our best)	**0.2072**
RAG + MS MARCO + Self-RAG re-ranking	0.1256

4.4 Model and Training Details

The router is trained with 15,000 examples per task category (datasets in Table 3) for one epoch, batch size 8, using a linear learning-rate schedule starting at 5×10^{-4}. Router accuracy is evaluated on held-out public data. All SFT and KD models are trained for three epochs on a single A6000 GPU using a cosine schedule with initial learning rate 1×10^{-5} and batch size 8. We fine-tune all parameters using AdamW with weight decay 0.05. KD uses temperature $\tau = 2.0$ and loss weight $\alpha = 0.8$. We compare against the zero-shot Gemma3-270M base model and the instruction-tuned Gemma3-270M-IT. Benchmark-specific inference hyperparameters follow Table 5. Full dataset details are provided in Appendix A.2.

4.5 Final Results

Table 2 reports the performance of our full system, which uses KD-both for reasoning, Data-Mixture-1 for instruction following, and RAG for factual QA (with IF as the fallback expert when routing is uncertain). The system delivers strong gains across all public benchmarks, surpassing the target scores for reasoning and instruction following and more than doubling zero-shot TriviaQA accuracy, demonstrating the effectiveness of combining modular experts with retrieval for small models. See Appendix A.5 for details.

Table 2: Final performance across all public benchmarks and the hidden evaluation set.

Model	**FactualQA (EM)**	**Reasoning (ACC)**	**Instruction Following**
Target	0.10	0.25	0.21
Base model (Gemma3-270M)	0.044	0.133	0.3465
Gemma3-270M-IT	0.103	0.266	0.3570
Our (public, w/ RAG)	**0.2072**	**0.2663**	**0.3923**
Our (hidden)	0.0900	0.2373	0.0460

5 Conclusion

In this project, we successfully fine-tuned Gemma-3 270M to meet or exceed the performance targets across multiple tasks. Our findings show that combining knowledge distillation, retrieval augmentation, and a modular expert–router architecture can substantially improve the abilities of very small language models. These results highlight the effectiveness of structured adaptation strategies in overcoming the limitations imposed by strict parameter budgets.

Author Contributions

The project was a collaborative effort by Group D. Z.J.: ablation study on hyperparameter choosing and training dataset composition, testing and analysis on router model, model integration, writing; E.J.: implementation of CoT and self-consistency, writing; B.K.: implementation and experiments with RAG, self-correction, DPO for reasoning and instruction following, training the model to use context (some experiments did not end up in the report), writing; T.L.: implementation, ablation studies, and analysis of router model, SFT and KD methods, writing; X.L.: develop the Self-RAG system, prepare the dataset for RAG, scoring model training, writing.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*, 2023. URL https://arxiv.org/abs/2310.11511. ICLR 2024.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Yelong Gao, Nan Duan Xiaodong Liu Jing, Tri Nguyen, and Rangan Majumder. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches*, 2016.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*, 2023.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022. URL https://arxiv.org/abs/2210.11416.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.14314.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Jack Rae, Tom Huang, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022. URL https://arxiv.org/abs/2203.15556.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yutaka Fujii, Alex Ratner, Ranjay Krishna, Ce Lee, and Thomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8003–8017, 2023. URL https://arxiv.org/abs/2305.02301.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. URL https://arxiv.org/abs/2106.09685.

HuggingFaceTB. Smoltalk: A collection of compact instruction and reasoning datasets. https://huggingface.co/datasets/HuggingFaceTB/smoltalk, 2024. Accessed 2025. Includes smol-constraints and smol-magpie-ultra subsets.

Mojan Javaheripi and Sébastien Bubeck. Phi-2: The surprising power of small language models. https://www.microsoft.com/en-us/research/blog/phi-2-the-surprising-power-of-small-language-models/, 2023. Microsoft Research Blog.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. URL https://arxiv.org/abs/2001.08361.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024. URL https://arxiv.org/abs/2411.15124. Includes the Tulu 3 instruction-following datasets used in this work.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, 2020. URL https://arxiv.org/abs/2005.11401.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, 2018. URL https://aclanthology.org/D18-1260.

NVIDIA. Openmathinstruct 2: A large-scale math instruction dataset. https://huggingface.co/datasets/nvidia/OpenMathInstruct-2, 2024.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2203.02155.

Post-training-Data-Flywheel. Autoif: Automatically generated instruction-following dataset. https://huggingface.co/datasets/Post-training-Data-Flywheel/AutoIF-instruct-61k-with-funcs, 2024.

Microsoft Research. Orca: Progressive learning from complex explanation traces. https://huggingface.co/datasets/orca_dataset, 2023.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009. doi: 10.1561/1500000019.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1701.06538.

Jiaxi Tang, Rakesh Shivanna, Zhe Zhao, Dong Lin, Anima Singh, Ed H Chi, and Sagar Jain. Understanding and improving knowledge distillation. *arXiv preprint arXiv:2002.03532*, 2020.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022a.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022b. URL https://arxiv.org/abs/2201.11903.

Wikimedia Foundation. Simple english wikipedia. https://simple.wikipedia.org/. Accessed 2025.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*, 2024.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024. URL https://arxiv.org/abs/2401.02385.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.

A Appendix

A.1 Model accessibility

The trained models are accessible from our code repository (we direct the readers to `README.md` for details). Both the model weights and training configuration files can be downloaded.

A.2 Datasets for Training

We train both the router LM and expert models using datasets covering reasoning, factual QA, and instruction following. Router training draws on ARC-Challenge and ARC-Easy (Clark et al., 2018) for reasoning, TriviaQA (Joshi et al., 2017) and OpenBookQA (Mihaylov et al., 2018) for factual QA, and instruction data from Tulu-3 (Lambert et al., 2024) and SmolTalk (HuggingFaceTB, 2024). These provide clear task boundaries required for reliable routing (Table 3). Expert SFT uses two mixtures combining Tulu-3 (Lambert et al., 2024), SmolTalk (HuggingFaceTB, 2024), and OpenMathInstruct-2 (NVIDIA, 2024) for instruction and reasoning, together with synthetic instruction datasets such as AutoIF (Post-training-Data-Flywheel, 2024) and Orca (Research, 2023). These mixtures enable controlled variation of task balance and model specialization under tight capacity limits (Table 4). Task-specific settings appear in Table 5. Full dataset details are provided in Appendix A.2.

Table 3: Summary of datasets for router LM training. Only the **train** splits were used for training.

Category	Dataset Identifier	Configuration/Subset
REASONING	allenai/ai2_arc	ARC-Challenge
	allenai/ai2_arc	ARC-Easy
	HuggingFaceTB/smoltalk	smol-magpie-ultra (reasoning)
INSTRUCTION	allenai/tulu-3-sft-personas-if	train
	HuggingFaceTB/smoltalk	smol-constraints
FACTUALQA	mandarjoshi/trivia_qa	rc.nocontext
	allenai/openbookqa	additional

Table 4: SFT mixture datasets

			# of samples	
Dataset Name	**Subset**	**Split**	**Data Mixture 1**	**Data Mixture 2**
allenai/tulu-3-sft-personas-instruction-following	—	train	Full	Full
HuggingFaceTB/smoltalk	smol-constraints	train	Full	Full
HuggingFaceTB/smoltalk	smol-magpie-ultra	train	60,000	60,000
nvidia/OpenMathInstruct-2	—	train_1M	10,000	30,000
Post-training-Data-Flywheel/AutoIF-instruct-61k-with-funcs	—	train	—	30,000
orca_dataset	—	train	—	30,000

Table 5: Hyperparameter settings by each task type.

Task Category	Max Tokens	Do Sample	Temperature	Top-p
Factual QA	64	True	0.0	1.00
Reasoning	512	True	0.7	0.95
Instruction Following	512	True	0.7	0.95
Default (Other)	512	True	0.7	0.95

A.3 RAG Prompt Template and Processor Details

This section details the exact prompt template and processing steps used by our `RAGProcessor`.

Prompt template. For a question q and its retrieved context C, the processor constructs the following zero-shot, retrieval-aware prompt:

```
Carefully use the following context to answer the question
correctly and factually.

Context:  C
Question:  q
Answer:
```

This prompt was shown to perform the best after some prompt engineering. If no context is available, the `Context` block is omitted and the prompt degenerates into a standard QA-style instruction of the form "`Answer the question.  Question:  q Answer:`".

Processor behavior. The `RAGProcessor` operates as follows:

1. Reads the `retrieved_context` field produced during offline retrieval.
2. Substitutes the raw text into the prompt template above (or into the fallback template if C is empty).
3. Wraps the prompt into the model's chat format using `apply_instruct_template`, treating it as a single user message.
4. Calls the base generation routine with zero-shot settings, typically using a small `max_new_tokens` budget for concise answers.

A.4 Details of the Self-RAG Style Re-Ranking Variant

In this section, we detail the implementation of the **Best-of-N** re-ranking strategy inspired by the Self-RAG framework Asai et al. (2023). This approach employs a specialized "Scorer" model to evaluate and select the highest-quality answer from multiple candidates generated based on retrieved contexts.

Data Construction. We utilized the open-source training dataset from the original Self-RAG project Asai et al. (2023), containing approximately 150k instances. To adapt this data for our scoring task, we performed an ETL (Extract, Transform, Load) process:

- **Format Transformation:** Original outputs were parsed to extract evidence paragraphs and critique tokens, specifically focusing on *Relevance* (e.g., `[Relevant]`), *Utility* (e.g., `[Utility:5]`), and *Support* (e.g., `[Fully supported]`). These were restructured into a standardized prompt format.
- **Negative Sampling:** To mitigate hallucination and improve discrimination, we augmented the dataset with negative samples. We replaced correct evidence with unrelated paragraphs and modified the target labels to `[Irrelevant]` with low utility scores, forcing the model to learn to reject unsupported answers.

Scorer Model & Training. We fine-tuned a `google/gemma-3-270M` model as the evaluator. The tokenizer was extended with special reflection tokens. Training was conducted in two phases:

1. **Phase 1:** Supervised fine-tuning on the processed Self-RAG data to learn the output format and critique criteria.
2. **Phase 2:** Refinement training using a balanced mix of positive samples and constructed negative samples to enhance the model's sensitivity to irrelevant context.

Inference Procedure. During inference on FactualQA tasks, the pipeline executes as follows:

1. **Expansion:** The retrieved context is split into chunks. The main QA model generates parallel candidate answers for each chunk.

2. **Scoring:** The Scorer model evaluates each (question, context, answer) triple. While the model is trained to predict support levels, for the final ranking score S, we focused on a weighted combination of Relevance probability and Expected Utility:

$$S = \alpha \cdot P(\text{Relevant}) + \beta \cdot \mathbb{E}[\text{Utility}] \tag{2}$$

   where $\mathbb{E}[\text{Utility}]$ is the expected value derived from the utility token logits (scaled to $0-1$). In our experiments, we set $\alpha = 0.4$ and $\beta = 0.6$.

3. **Selection:** The candidate with the highest score S is selected as the final prediction.

Observed Limitations. We observed that the small-scale Scorer (270M parameters) effectively identified relevance but occasionally struggled with subtle factual hallucinations compared to larger judge models. Nevertheless, the pipeline successfully filtered out obviously irrelevant answers generated from noisy retrieval contexts.

A.5 System Wiring, KD Settings, and CoT Augmentation (Reproducibility)

Specialist wiring and router checkpoints. Our submission wires a router-based pipeline that dispatches inputs to task-specialized checkpoints (configured in `submit.py`):

- **Instruction specialist (Mixture-1)**: `experiments/sft_only_tulu3_smoltalk`
- **Reasoning specialist (KD-both v2)**: `experiments/sft_kd_both_v2`
- **QA specialist (base model with RAG)**: `google/gemma3_270m`
- **Router (Gemma3-270M)**: `experiments/router_15k_gemma3_270m`

These paths match the final submission pipeline. The router predicts one of {factual QA, reasoning, instruction following} and selects the corresponding specialist.

KD hyperparameters. To aid reproducibility, we surface key KD settings used in the best reasoning specialist:

- Distillation weight $\alpha = 0.8$
- Teacher temperature $T = 2.0$
- Max sequence length $L_{\max} = 768$

These govern the supervised vs. teacher blend, the softness of targets, and the truncation window during training.

CoT augmentation workflow. We augment QA/reasoning data with chain-of-thought (CoT) rationales using a teacher model:

1. **Teacher prompting**: Inputs are expanded with CoT prompts; the teacher outputs an "analysis" (reasoning trace) and a final answer.
2. **Insertion format**: A `### Analysis` block is inserted into the sample. The order is controlled by `cot_reasoning_first`:
   - `true`: rationale before the answer;
   - `false`: answer before the rationale.
3. **Sampling**: Augmented instances are mixed with originals according to `augmentation_ratio` to avoid overfitting to long traces.
4. **Outputs**: Augmented files are saved to `./augmentation` and consumed by SFT/KD.

Factual QA model choice. For factual QA, we do *not* use a QA-tuned specialist. The final system employs the **base Gemma-3-270M** paired with retrieved contexts (BM25 RAG). Performance gains come from retrieval conditioning, optionally with the Self-RAG scorer for re-ranking.

A.6 METHODOLOGICAL INSIGHTS

Our system focuses on clear methods, not just scores.

- We use a router to send inputs to small specialist models. This reduces forgetting and keeps each model focused on one job.
- For reasoning, we combine soft-logit distillation and chain-of-thought supervision. Logits teach the distribution. CoT teaches the reasoning steps. Together they work better than either alone.
- For factual QA, we use the base Gemma-3-270M with retrieval. External memory is more reliable than trying to store facts in a small model.
- We tune three key hyperparameters: α, T, and $L_{\max}$. These control stability and quality. Documenting them makes our work reproducible.
- We control how we add CoT. The order of the rationale and the sampling ratio matter. These choices change how a small model learns.
- We prefer simple methods when complex ones add noise. Self-RAG re-ranking added variance and missed subtle errors. BM25 retrieval alone was more reliable.

This design is a practical template for small models. It shows how routing, KD, SFT, and RAG fit together. It exposes the key knobs so others can reproduce and improve the method.