NLP Extended Task

Lee Sean

1006903@mymail.sutd.edu.sg

1. Introduction

The evolution of Large Language Models (LLMs) has traditionally been driven by scaling parameter count and training data, evident by the architectural progression from the initial Transformer architecture, to larger and denser models like GPT2. For a period of time, this became the dominant strategy for development, but it came at the cost of unsustainable computational demands that scaled linearly with model size. This bottleneck caused the field to undergo a paradigm shift by reducing the proportion of parameters that are active per input but still maintaining performance, opening a new pathway for model optimization and ultimately the resurgence of the Mixture of Experts architecture, albeit the fact that it was a modernized version from the initial model proposed by Jacobs et al. (1991). This modern implementation involves a router dynamically activating parts of the model, envisioned as experts, to process data in a computationally and sample efficient manner. With these ideas, this report aims to explore the implementation of the MoE architecture in GPT2 for cross-lingual transfer and draw comparisons to previous tasks.

2. Related Works

2.1 Sparsely-Gated Mixture-of-Experts Layer

Shazeer et al. (2017) introduced the idea of a sparsely gated network MoE within Long-Short Term Memory (LSTM) Recurrent Neural Networks (RNNs), allowing for the ability to make discrete selections, which was a departure from the initial idea of MoEs taking a softmax weighted value of all the experts resulting in a dense model approach. With this implementation, each expert would be dynamically activated based on the discrete choice, saving resources by reducing the proportion of parameters that are active at each time. To further improve the learning of this discrete selection, the authors opted to add randomness, called Gaussian noise, to the scores for each expert, helping with load balancing. Adding randomness incentivises the router to explore, by occasionally allowing experts that are just barely out of the top-k experts to get selected for activation. This prevents the scenario where an expert with a slightly better initialization is repeatedly chosen over another expert that may be the better choice in the long run. However, this was not enough to stop the model from having the tendency to produce larger weights for the same few experts, so it was used in conjunction with an additional loss term, $L_{importance}$ that summed the probabilities of each expert for that batch, and then penalised the model by how unequal the distribution is by weighting the square of the Coefficient of Variation.

2.2 Switch Transformer

Fedus et al. (2021), although not the first encoder-decoder model to include the MoE architecture, marked a critical divergence from previous works like Gshard (Lepikhin et al., 2020) by simplifying the routing. The authors demonstrated this by using a top-1 routing strategy as opposed to the previously standard top-2 expert strategy, quantifying their results by showing that their architecture had faster convergence time compared to previous MoE baselines. Moreover, they adopted Gshard's auxiliary loss and random fixed noise (router jitter) formulation over the more complex importance loss used by Shazeer et al. (2017). Auxiliary loss was introduced as an elegant way to add a penalty term to the model's training objective to force distribution of tokens evenly across all experts. It functions by minimizing the sum of the dot product between two terms: usage (the fraction of tokens that actually went to the expert in the batch) and probability (how confident the router was in sending the tokens to that expert) (Lepikhin et al., 2020).

2.3 Mixtral of Experts
Jiang et al. (2023) presented a highly capable open-source sparse MoE decoder-only LLM that was relatively small at only 47 billion parameters. Through the development and experimentation of this new model, the authors were able to prove that a smaller sparse MoE model was able to perform on par with larger dense frontier models, all while substantially improving computational efficiency. This solidified MoE as a viable technological advancement in LLM design.

2.4 Upcycling LLMs into MoE
Building on the concept of sparse upcycling from Komatsuzaki et al. (2022), He et al. (2024) showcased a modern methodology to upcycle decoder-only dense models into the industry standard MoE architecture. According to the authors, the motivation behind this research was to leverage existing dense frontier models and convert them into an MoE architecture, reducing the computational overhead of retraining the architecture from scratch, and establishing a pathway for improving the accuracy of existing models while adhering to the computational constraints. Technically, this was achieved by converting the feed forward network (FFN) in the dense model into a sparse MoE layer, replicating the FFN weights by a predefined number and adding a randomly initialized router layer. To break the symmetry of the experts initialized with replicated weights, a virtual group is used. However, as this specific technique is optimized for upcycling involving multiple MLP shards, it falls outside the scope of this report and will not be discussed further.

2.5 Drop Upcycling
Recognizing the potential pitfalls of expert collapse[1] and slower convergence, Nakamura et al. (2025) adapted the aforementioned upcycling method by introducing 'Drop-Upcycling'. This framework included a more aggressive reinitialization strategy of the experts: statistical reinitialization of a predefined proportion of each expert's weights. This statistical reinitialization aims to find the balance between knowledge retention and expert diversification by using a normal distribution centered at 0 with a standard deviation scaled to match the original dense weights. This leads to a starting model with structurally distinct experts, therefore providing the router with distinguishable signals that naturally encourage diversification as early as the first training step.

3. Experiment
Based off the literature review from the previous section, the final setup of our upcycled MoE architecture is as follows:

1) Auxiliary loss
   Based off Gshard (Lepikhin et al., 2020) and Switch Transformer (Fedus et al. 2021), we implement the auxiliary loss weighted by a predefined load balance weight, allowing fine grain control over how much the auxiliary loss affects the overall penalty of repeatedly choosing the same expert. Through this, we aim to promote expert diversification and lead to faster training convergence.
2) Router Jitter
   Learning from Gshard (Lepikhin et al., 2020) and Switch Transformer (Fedus et al. 2021), a fixed amount of random noise is introduced to the logits produced by the router, allowing for expert diversification and exploration during training.
3) Expert Reinitialization Strategy
   Drawing inspiration from the drop-upcycle methodology (Nakamura et al. 2025), the proposed model also adopts reinitialization of a proportion of each expert's weights (referred to as drop ratio), aiming to achieve better performance in an efficient amount of training steps.
4) Interleaved MoE layers
   Following the recommendation from Switch Transformer (Fedus et al. 2021), we choose to apply the MoE layer to every other layer in the architecture, allowing for a smaller model overall since there are less parameters without sacrificing too much performance.

[1] *Expert collapse* refers to the phenomenon where the router favours one expert over the others, where the 'rich get richer' and the router learns to depend on only one expert, effectively regressing to a smaller dense model where all inputs flow through one expert and the others are left inactive.

3.1 Configuration

1) Top-k Expert: 1
   To ensure a fair comparison with our dense baseline, we maintain the same floating point operations per second (FLOPs) as GPT2, by only allowing the router to choose 1 expert. This choice aligns with the Switch Transformer architecture (Fedus et al. 2021), which demonstrates that choosing only 1 expert is a viable option for the MoE architecture.
2) 8 Experts
   Following the footsteps of the high performing Mixtral of Experts model, we adopt a total of 8 experts, providing a balance between sparsity and capacity to specialize.
3) Reinitialization Ratio: 0.15
   While the drop-upcycling paper (Nakamura et al. 2025) has results that show having a reinitialization ratio of 0.5 produces the best result, the knowledge loss that is incurred would be unfeasible for our experimentation because it requires massive amounts of training data to recover the lost weights. Given our computational constraints, we chose a more conservative 0.15 reinitialization ratio, aiming to attain results that are on par with or are better than our dense baseline even with the limited training budget.
4) Load Balance Weight: 0.01
   Building off the findings from the upcycling paper (He at el. 2024), we chose a load balance weight of 0.01, because they found that values ranging from 0.01 to 0.001 produced the best language model loss when applied to the auxiliary loss.
5) Effective Batch Size: 256
   Finally, the upcycling paper (He at el. 2024) found batch sizes of 512 and 1024 being ideal for convergence. However, due to GPU memory limitations, we opted to implement the closest effective batch size possible that our hardware infrastructure was able to support.

3.2 Datasets and Metrics

For continued pretraining of the upcycled GPT2 MoE architecture, we use C4, a cleaned version of Common Crawl's web crawl corpus. This dataset was intentionally chosen for its similarity to gpt2's original dataset, which uses WebText dataset, so that we can try to reduce variance in training data.

For cross-lingual transfer training and evaluation, we use XNLI-MT, an evaluation corpus for language transfer and cross-lingual sentence classification in 15 different languages. This allows us to measure performance over a range of languages and give us control over what kinds of language we want to evaluate on.

	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh
Fertility Score	**1.11**	4.70	5.53	2.10	6.16	**1.83**	**1.75**	5.12	5.90	2.08	9.48	2.73	5.16	3.62	3.77

Table 1: Fertility score of the available languages on XNLI-MT dataset

For both, one-shot and few-shot experiments, we chose languages that had a low fertility score. Fertility measures the average number of tokens produced per word, thus indicating a good measure for how well a specific tokenizer can carry over to other languages. Since our model uses an English tokenizer, we choose to take languages closer to the fertility score of English, which is 1.11, to be able to see tangible performance gains that may have otherwise been masked by poor tokenization. Thus, we take French as our one-shot cross-lingual transfer, with the next lowest fertility score at 1.75, and add on Spanish for our few-shot cross-lingual transfer.

For both tasks, we choose to use accuracy as an easy to compute baseline judge to quickly evaluate the comparative performance of the vanilla GPT2 architecture and the drop upcycled MoE architecture.

3.3 Continued Pretraining Results

Due to computational limits, we only managed to train at an effective batch of 256 (batch size of 64 and gradient accumulation step of 4) for a total of 14649 training steps, utilizing around 3.75 million data samples from C4, roughly 1.6 billion tokens[2]. After continued pretraining on the drop upcycled MoE weights, we obtained an accuracy of 38.23% and a validation loss of 3.4 on the C4 dataset.

3.4 Cross Lingual Transfer Results

Table 2 shows the accuracy on our model after being finetuned on the French dataset. As seen above, our model outperforms GPT2 on the French test datasets, while the results on the zero-shot cross lingual transfer tasks stayed relatively random.

	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh
MoE	56.01	36.73	37.11	45.41	**35.65**	**52.93**	**64.91**	**36.65**	36.65	40.92	**36.99**	**42.30**	**36.53**	**40.76**	39.08
GPT2	**57.15**	**37.09**	**37.76**	**45.53**	36.47	52.53	63.79	36.35	**36.89**	**43.29**	35.13	41.98	34.43	39.50	**43.43**

Table 2: Accuracy for one-shot cross lingual transfer on French

Table 3 shows the accuracy on our model after being finetuned on the English, Spanish and French dataset. The results prove that our model yields better results on only the Spanish test datasets, while the results on the zero-shot cross lingual transfer tasks remain relatively random.

	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh
MoE	75.31	35.11	34.75	**47.05**	35.53	**67.11**	65.49	36.19	34.70	41.22	35.71	**42.20**	34.33	36.11	35.11
GPT2	**76.51**	**37.60**	**37.33**	44.49	**37.82**	66.25	**66.71**	**39.04**	**38.12**	**41.54**	**39.82**	42.16	**37.47**	**39.06**	**38.02**

Table 3: Accuracy for few-shot cross lingual transfer on English, French and Spanish

Analysis

Our experimental results show that our developed MoE architecture outperforms the original GPT2 model on both one-shot and one of the few-shot cross lingual transfer tasks (Spanish). Despite only being able to beat GPT2 in Spanish, it is possible that with more training of the router and experts, our model has the capacity to surpass GPT2 in few-shot cross lingual tasks since it has already shown the ability to do so for French in the one-shot task. Notably, this conclusion was drawn with limited pretraining data and clear indications of the infancy of our developed model. From the upcycling paper (He et al. 2024), we can see that performance between continued training and upcycled models only become more apparent after the models have been exposed to a few billion tokens. Moreover, adopting the drop-upcycling method (Nakamura et al. 2025) incurs an additional adaptation cost to recover from the reinitialized weights, requiring even more pre-training data. Despite this, our model was able to generate comparable accuracy to the original GPT2 model while only conducting continued pretraining on roughly 4% of the original training data. This trajectory indicates significant potential for the upcycled model, especially given the sustained downward trend in the validation loss observed towards the end of the training period.

An analysis of the expert distribution shows mixed results, between task-agnostic generalization and language specialization. On one hand, we see full generalisation in terms of classification labels (*Figure 2* & *Figure 4*), where for every expert, the amount of tokens routed remain uniform across entailment, neutral and contradiction labels. This implies that the model does not route based on high-level semantic understanding but rather low-level syntactic features. Conversely, certain experts show specialisation when it comes to languages. This is evident in the one-shot cross lingual task, with expert 5 having a higher preference for English tokens while expert 0 shows a higher preference for French tokens (*Figure 1*).

[2] due to the limit on the number of hours a colab pro instance can run for on google colab (24 hours), we split the training into 2 parts: Firstly, an effective batch of 254, 4883 training steps, dataset shuffle seed of 42 and secondly, an effective batch of 254, 9766 training steps, dataset shuffle seed of 1337

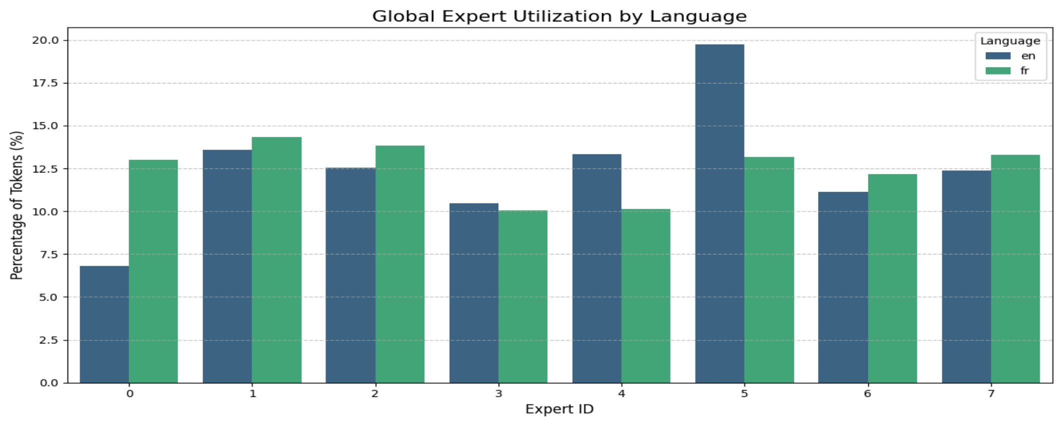

Figure 1: Expert distribution based on language for one-shot experimentation

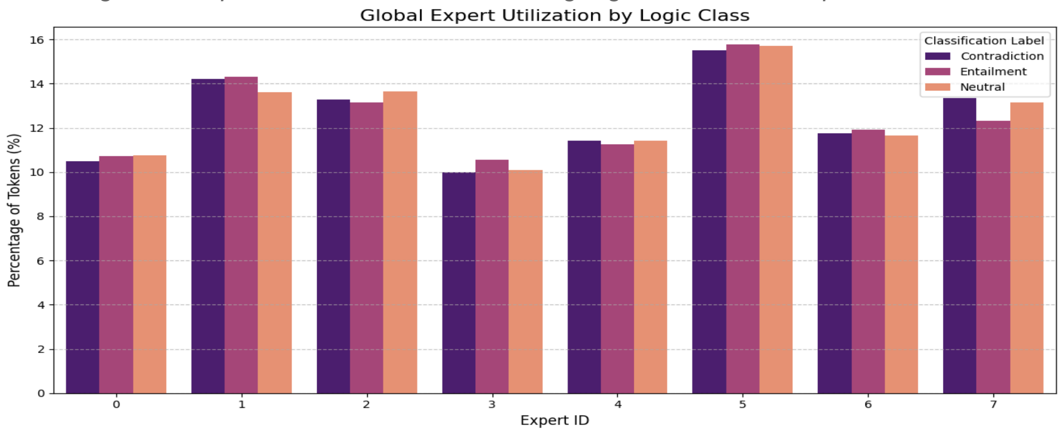

Figure 2: Expert distribution based on label for one-shot experimentation

However, this specialization destabilised during few-shot transfer. We propose that the decline in French performance during the few-shot transfer stems from a critical dilution and redistribution of data across the expert layers. Expert 0, which was the primary French specialist in the one-shot experiment now exhibits a competing preference for Spanish (*Figure 3*), where the limited capacity of the expert was forced to accommodate both French and Spanish syntax, affecting specialization. Extending from the increased workload of expert 0, the router was forced to push other generalized experts to specialize, resulting in experts 6 and 7 beginning to favour non-english tokens as well. While this leads to a more diverse pool of experts, the result is a more distributed flow for French tokens across the 3 distinct experts. This reduces the effective training sample size seen by the supposedly highly specialized experts leading to underdevelopment of expert 0 compared to the more rigorous training in the one-shot task. We posit that this observed degradation can be resolved through a more optimized router directing specialized tokens to the respective specialized experts or a more robust routing-strategy like Top-2 routing, for greater involvement of the experts.

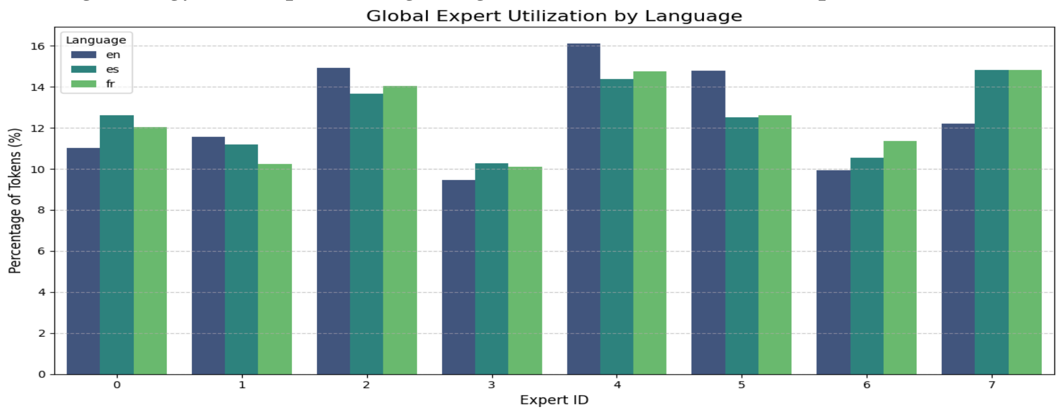

Figure 3: Expert distribution based on language for few-shot experimentation

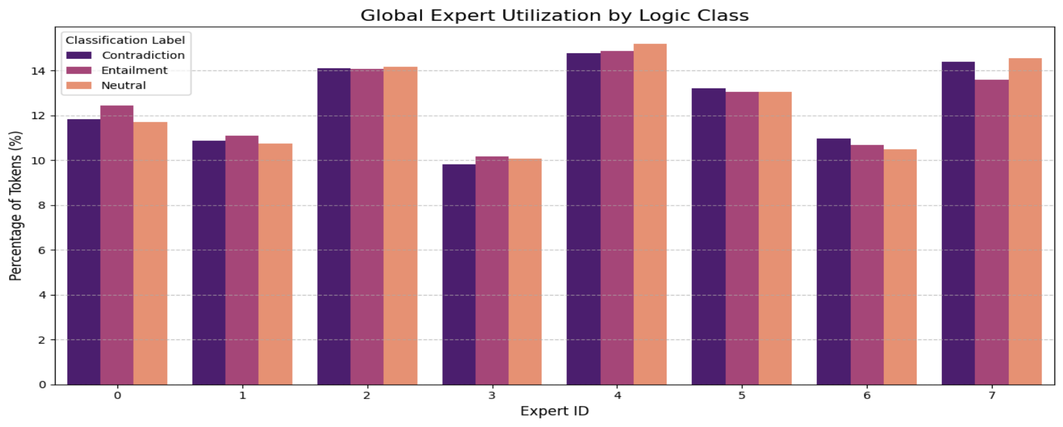

Figure 4: Expert distribution based on logic for few-shot experimentation

Future Works

Since this experiment was approached from the angle of exploration, there are areas that require greater scrutiny to be able to definitively conclude the claims made.

1) Equal Baseline
   To construct a truly fair comparison, we need to allow the original GPT2 to be tuned on the C4 dataset. We propose performing continued pretraining on the same training configurations to the original GPT2 that the upcycled model was exposed to. After this, a comparison on the one-shot and few-shot cross lingual transfer tasks can be run without fear of training variations.
2) Different Hyperparameters
   To better gauge the robustness and performance ceiling of the upcycled GPT2 architecture, a broader range of tests could be run. A potential direction would be developing 3 models of increasing performance and running benchmarking against its original dense counterpart. This direction allows us to isolate architectural changes, evaluating the advantages of MoE even under sub-optimal configurations and enhancements that could resolve the observed dilution in few-shot cross-lingual transfer tasks.

The exploration detailed in this report opens up new avenues of experimentation that motivate the further development of the upcycled MoE architecture.

1) High Fertility Languages
   One such approach would be to further run fine-tuning and test on languages that have a higher fertility score. Through this exploration, we propose aiming to see if the MoE architecture is able to make up for the poorer tokenization of data, using GPT2 as the baseline.
2) Different Variations
   Finally, we propose further iterative improvements on our upcycled architecture through expert partitioning, as seen in DeepSeekMoE (Dai et al. 2024) and Mixture of Million experts (He 2024). This deep dive into the variations of MoE will allow us to not only improve on the current model, but expose the idea behind how introducing even more sparse experts can further push performance to the next level.

References

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, Geoffrey E. Hinton:
Adaptive Mixture of Local Experts (1991)

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, Jeff Dean:
Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. CoRRabs/1701.06538 (2017)

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, Zhifeng Chen:
GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding. CoRRabs/2006.16668 (2020)

William Fedus, Barret Zoph, Noam Shazeer:
Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. CoRR abs/2101.03961(2021)

Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, Neil Houlsby:
Sparse Upcycling: Training Mixture-of-Experts from Dense Checkpoints. CoRR abs/2212.05055 (2022)

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, William El Sayed:
Mixtral of Experts. CoRR abs/2401.04088 (2024)

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, Wenfeng Liang:
DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models. CoRRabs/2401.06066 (2024)

Xu Owen He:
Mixture of A Million Experts. CoRR abs/2407.04153 (2024)

Ethan He, Abhinav Khattar, Ryan Prenger, Vijay Korthikanti, Zijie Yan, Tong Liu, Shiqing Fan, Ashwath Aithal, Mohammad Shoeybi, Bryan Catanzaro:
Upcycling Large Language Models into Mixture of Experts. CoRR abs/2410.07524 (2024)

Taishi Nakamura, Takuya Akiba, Kazuki Fujii, Yusuke Oda, Rio Yokota, Jun Suzuki:
Drop-Upcycling: Training Sparse Mixture of Experts with Partial Re-initialization. CoRR
abs/2502.19261 (2025)