Exploring Mixture of Experts and Sparse Vision Transformers

Student Name: Christopher Teo
Supervisor Name: Professor Paolo Remagnino
Submitted as part of the degree of BSc Computer Science to the
Board of Examiners in the Department of Computer Sciences, Durham University

Abstract—**Context:** With the increasing size of models with billions of parameters, a class of transformers has emerged to become one of the hottest topics in machine learning: mixture of experts (MoEs). MoEs have been dominating the natural language processing space with their ability to be trained with far less computation than their dense counterparts and have been revisited for computer vision applications.

Aims: To investigate how best to apply MoE architecture, combining recent advances, for image classification. This investigation will focus particularly on how the experts specialize and how the router performs task decomposition for said experts.

Method: This paper introduces a solution for overcoming the non-differentiability problem with sparse MoEs through the use of transfer learning. First, a fully dense MoE is trained and then is switched to a sparse MoE whilst retaining the expert weights and only re-training the new sparse gate. Furthermore, a unique approach for load balancing experts is introduced in a way that does not hamper their ability to produce specializations by encouraging variance within the gate outputs for a given batch.

Results: The final model is able to beat all previously surveyed techniques and is able to converge on a very intuitive expert specialization scheme that aligns with human perception.

Conclusion: The current approach to load balancing MoEs may be holding their performance back as they encourage poor expert specialization. Rather than trying to balance for expert utilization, models should be aiming for maximum variation in expert selection across a dataset to motivate highly semantic specializations.

Index Terms—Machine Learning, Deep Learning, Computer Vision, Mixture of Experts, Sparse Vision Transformers

1 Introduction

Image classification has always been a large application of Machine Learning (ML), wherein some Deep Learning model (classifier) is designed to correctly classify a set of images to their respective classes. Traditionally, prior Deep Learning (DL), this was done using hand-crafted algorithms through various image processing techniques such as autothresholding [1] to extract manually selected features which could then be used with an ML classifier such as Support Vector Machines (SVM) [2] or Random Forests [3]. With the introduction of DL, feature extraction could now be done by the model in which the model is trained using a large dataset of images to learn what features to extract. EarlyDL models used for image classification were convolutional neural networks (CNN) [4] which far outperformed its competition at the time. This is because, unlike other models, CNNs are able to leverage the spatial coherence within images, exploiting the fact that neighboring pixels / regions of pixels are inherently related allowing them to extract meaningful features much more easily.

Recently, transformer models [5], [6], first introduced in natural language processing (NLP), have been steadily gaining prominence over CNNs with their ability to capture global relationships between far pixels / regions unlike CNNs that only capture local features. Transformer models come at the cost of being slightly more expensive to compute as well as requiring far longer training times as they must learn these relationships, in contrast to CNNs which are locally biased by design through its use of convolutions which can only act upon local pixels [6]. For this reason, it is common to have transformers pretrained such that they can learn relationships prior to being applied to downstream tasks such as classification. A common task used for pretraining is image reconstruction through which the transformer is used as the encoder for an auto-encoder model to reconstruct parts of an image [7]. The encoder can then be used as part of a different model to be trained for a particular downstream task. This is referred to as fine-tuning.

Recently, as these models have continued to grow in scale, there has been a reemergence of a certain architecture commonly known as mixture of experts (MoE), most notably in NLP with the majority of their top performing models such as Mistral 7B [8]. At a higher level, this refers to combining multiple specialized models, or "experts", to collectively solve complex tasks. This is often done with the use of a gating model which aims to route a given input through the best suited experts. This can involve passing an input through all experts and performing some weighted combination of the outputs based on the gate or by only passing the input through a small subset of experts. Currently, there are 2 main types of MoEs:

- **Sparse MoE** - Only some experts are utilized when generating an output.
- **Dense MoE** - All experts are utilized when generating an output.

Although on the surface this may seem like a distinction rather than a difference, this actually has large implications for the model. In particular, sparse MoEs have the following properties:

- **Conditionally computed (Sparsity)** [9] - As only a subset of the model is required for a given output, compute can be saved by not computing the output of unused experts.
- **Expert imbalance** - Updating only the selected experts for each training sample results in undertraining of other experts and reinforces a fixed selection by the router, hindering proper specialization.

On the other hand, dense MoEs are:

- **Fully differentiable** - As the output relates to all experts, the entire network is differentiable for a given input. This allows back propagation throughout the whole network for a given input circumventing the issue of expert balance with sparse MoEs.

The aim of this paper is to investigate the use of MoE for image classification in computer vision using the CIFAR100 dataset by combining some of the latest state of the art (SOTA) MoE techniques. Accompanying this is a secondary goal of trying to combine techniques between sparse and dense models to overcome the weaknesses of both approaches. Primarily, this study will focus on the routing strategies these models learn and how various techniques affect how well the model is able to decompose the given problem among its experts. Throughout this paper, this will be referred to as task decomposition.

2 Related Work

The idea of an MoE was first introduced by Geoffrey Hinton [10] which proposed a supervised learning framework involving an ensemble of several models. Each model would be trained to process a subset of the training data, and a single gating model would then be trained to select an expert based on the input describing the first sparse MoE architecture. Since this proposal, several works have explored this idea of conditional computing but despite this, MoE still remained a niche idea yet to be adopted for the following reasons:

- **GPU hardware** - MoE models of this era were not well suited for GPUs as they could not easily be parallelized due to their sparse nature as GPUs prefer to execute many identical operations concurrently as opposed to being conditional.
- **Dataset sizes** - The benefits of MoE do not really apply to the small datasets of the time as these could be learned by smaller dense models much more efficiently.
- **Large batch sizes** - Significantly larger batch sizes are required to train MoE models as each expert is only exposed to a subset of each batch.
- **Task Decomposition** - It was up to a human to determine how a given task should be divided among the experts. This meant knowledge of the dataset and task was required making the mode unfit for other tasks, especially ones with non-trivial decompositions.

Since then, the amount of available compute, size of datasets and scale of models have dramatically increased, facilitating the resurgence of MoEs. In particular, Noam Shazeer et al. [11] introduced the idea of embedding MoE within a model as a single MoE Layer which they applied to a long short-term memory (LSTM) model for NLP. In this architecture experts are no longer separate models that would be trained separately but rather were single modules within a given layer. Through this, all experts, including the gate, could be trained end-to-end removing the need for a manually designed task decomposition as the network could learn one itself. Unfortunately, it was common for the gating mechanism to converge to a state that always selected the same experts for every input. This phenomenon stemmed from the fact that selected experts were trained more rapidly than others and thus the gate is incentivized to continue selecting the same trained experts. To mitigate this, an additional auxiliary loss term was added to act as a "soft" constraint, enforcing that the gate selects experts equally within a given batch. Additionally, it was hypothesized that it was necessary to route the input through at least 2 experts. The intuition behind this was that at least 2 experts were needed with each forward pass to be able to effectively learn which experts should be selected as the gate would be provided with non-trivial gradients.

The work done by Fedus et al. [12] and Dmitry Lepikhin et al. [13] extended this concept to the more recent transformer architecture with the switch transformer and GShard framework. This was done by replacing the multilayer perceptron (MLP) layer of the transformer [5] with an MoE layer as described previously. Through this, each token of the transformer is passed through the MoE layer separately and thus tokens in the same sequence are not always sent to the same experts. Additionally, they found that it was actually not necessary to require at least 2 selected experts for effective training of the gate. As the switch transformer was distributing each token for a given input, it was important to also balance the distribution of tokens among the experts. To this end, they added a term to the auxiliary loss used previously to account for token distribution as well as expert utilization. As well as this, they also implemented a capacity factor which determines how many tokens an expert is allowed to process per batch. If a token is routed to an expert which has hit its max capacity, the token is simply skipped and dropped. It should be noted that this capacity factor must be carefully tuned to avoid having too many dropped tokens.

Interestingly, the use of a capacity factor has the additional benefit of being able to set the total capacity of all experts below the total number of tokens which was explored by Riquelme et al. [14] in which they used a low capacity factor to further reduce compute. To ensure that only the most important tokens were processed with the limited capacity, they utilized the gating weights to assign a priority to each token such that tokens weighted more heavily for a given expert have priority over those of lower weighting. Through this they found that MoEs for image classification are quite robust to low capacity factors and can perform

on par with models without limiting capacity factors with only 20-30% of tokens being processed. Furthermore, this limitation can be applied after training making it incredibly flexible as a model does not have to be pretrained with a low capacity factor and thus can be applied to many models out of the box.

So far, many of these works have addressed the problem of balancing tokens across experts through the use of an auxiliary loss function. This was because they relied on tokens choosing top-k experts to be routed through. On the other hand, the experts could instead choose top-k tokens to process as proposed by Yanqi Zhou et al. [15]. This has the benefit of guaranteeing a balanced distribution of tokens among all experts as well as offering varied compute per token as a token can be passed through multiple experts as opposed to just 1. The distinction between tokens choosing top-k vs experts choosing top-k is often referred to as token choice vs expert choice, and it was observed by Tianlin Liu et al. [16] that expert choice outperforms token choice algorithms. It should be noted that expert choice utilizes both future and past tokens unlike token choice algorithms which is problematic for tasks such as autoregressive text generation. Fortunately, for the task of image classification, this does not pose as an issue.

Across the majority of the surveyed papers, a key aspect has been the balanced distribution of tokens among experts. It should be noted that this enforces a particular type of task decomposition in which experts are heavily encouraged to specialize in ways that can be distributed among the fixed expert count. This may harm the overall performance of the model depending on the task and expert count if there does not exist an efficient task distribution that utilizes all available experts equally. It is for this reason that routing algorithms such as expert choice [15] are avoided as they hard enforce equal token distribution by design. This paper proposes a different soft constraint that does not enforce an equal distribution of tokens but still enforces variation in the selected experts.

Currently, many transformer models implement the MoE layer as the MLP layer of the transfer that appears after multi-headed attention [5]. Róbert Csordás et al. [17] proposes a different approach in which the MoE layer is applied to the attention mechanism in which experts project query, key and value vectors in order to reduce the number of attention heads. This is because the computation of projections through experts scales linearly whereas the computation of attention heads scales quadratically. Additionally, this can be combined with MoE in the MLP layers to further decrease computation. Unfortunately, this paper will only focus on MoE within the MLP layer due to the lack of compute resources required for large MoE models because, despite lowering compute, MoEs require all experts to be reserved in memory. To add to this, architectures using hierarchical MoEs [18], in which routers are stacked creating a set of MoEs to choose from, are also avoided due to this limitation.

In the majority of surveyed work, as stated by Joan Puigcerver et al. [19] the issue of MoE models being non-differentiable is rarely addressed. This is a problem as it is because of this fact that sparse MoEs suffer from expert imbalance, in particular the issue of under trained experts as a result of them not being utilized in the forward pass alongside gating networks not being able to relate to all experts for a given input sample in training. Joan Puigcerver et al. proposes a unique approach to MoE which foregoes the benefits of conditional computation in favor of being fully differentiable by remaining a fully connected model. At a high level, this is achieved by having the experts use a weighted sum of all tokens as opposed to only selecting the top-k tokens. Despite being a fully connected model, they argue that the model still follows MoE design philosophy as each expert only views a subset of the tokens via the weighted sum.

3 METHOD

3.1 Background Theory: MoE Transformer

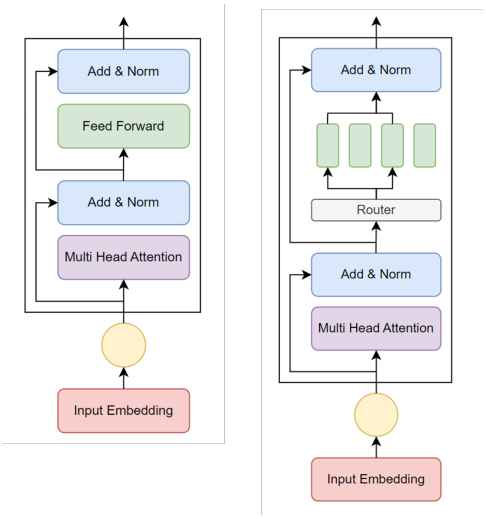

Fig. 1. Standard transformer architecture (left), MoE transformer architecture (right).

The standard dense transformer consists of multi-headed attention and a dense fully connected feed forward layer, commonly referred to as the MLP layer, as shown in the above Figure 1. The MoE layer replaces this MLP block in the standard transformer to produce an MoE transformer.

This MoE layer consists of some gate (router in Figure 1), G, and some set of experts, $E : \{e_i\}$, where i denotes the ith expert. The output for a given MoE layer is then defined as follows:

$$\text{MoE}(x) := \sum_{i=1}^{|E|} G(x)_i \cdot e_i(x)$$

Where G and e_i are functions yet to be defined.

3.2 Architectural Overview and Design Choices

For pretraining, a standard masked auto encoder (MAE) [7] setup will be used as it is the current SOTA method for pretraining vision transformers. The input to the model will consist of 36x36 images from the CIFAR-100 dataset. This is done by slightly upscaling the images from 32x32 which is done because Jen Hong Tan [20] showed that upscaling the images to accommodate more tokens with the use of 3x3 patches improves performance greatly as there are 144

TABLE 1
Parameters for pretraining

Parameter	Value
Max epochs	3000
Patch size	3
Embed dim	192
Num experts	4
Depth	6
Num heads	12
Decoder dim	192
Decoder depth	4
Decoder heads	6
Mask ratio	0.75
Batch size	1048
Base lr	1.5e-4
Optimizer	AdamW

TABLE 2
Parameters for finetuning

Parameter	Value
Max epochs	200
Patch size	3
Embed dim	192
Num experts	4
Depth	6
Num heads	12
Batch size	768
Base lr	1e-3
Optimizer	AdamW

total tokens as opposed to the traditional 64 from the use of 4x4 patches with 32x32 images. Additionally, the CIFAR-100 dataset would be auto augmented [21] following the CIFAR-10 policy.

As for the MoE layers, they were applied to all transformer layers within the encoder of the model. This is because this paper only focuses on the performance of MoE for image classification in which the encoder is the only part of the MAE used for the downstream task. The full recipe for pretraining can be found in Table 1.

Within the MoE layer, the experts, e_i, all consisted of a single MLP layer. As for the gate, another single MLP layer is used who's outputs are passed through a softmax function to provide weights describing how the experts should be combined to generate the outputted tokens, x':

$$x' := \sum_{i=1}^{|E|} G(x)_i \cdot e_i(x) \text{ with } G(x)_i := \text{softmax}(Wx + \epsilon)[i]$$

For finetuning, the encoder of the MAE model is isolated, and a classification head is appended to the CLS token of the transformer [6] to perform the actual classifications. The classification head simply consists of a single MLP layer outputting classification probabilities for each class of image. The full recipe for finetuning can be found in Table 2.

Currently, this setup is in the form of a dense MoE such that during pretraining and finetuning the experts are fully differentiable circumventing the shortcomings of sparse MoEs in which there is imbalanced training of experts. However, in order to still reap the benefits of conditional computation at inference time, the model is further finetuned with its dense MoE layer replaced with a sparse MoE layer in which only the top-k expert from a gate is used:

$$G(x)_i := \text{top}_k(\text{softmax}(Wx + \epsilon)[i])$$

During this process, the entire model, except for the gates, have their weights frozen such that only the new gates are being trained to perform the same task decomposition and routing as the previous gate. Through this, the knowledge of the old dense gate is distilled into the sparse gate, maintaining its learned task decomposition whilst now being able to take advantage of conditional computation. Additionally, to accommodate for the fact that the previous model has additional signals through its utilization of all experts, after distillation, the whole model is unfrozen and allowed to finetune freely to make up those losses post training the sparse gate.

To aid with this, some soft constraints need to be added to the dense gate such that it generates a learnable output for the sparse gate. Firstly, the output of the dense gate should approach a one hot vector as the sparse gate acts affectively as one thus making it difficult for the sparse network to match performance if the dense gate heavily relies on more than 1 expert for a given output. To enforce this, a soft constraint describing that the inverse coefficient of variance, $CV'(x)$, of the generated expert weights should be as small as possible was added:

$$CV'(x) := \frac{\mu(G(x))}{\sigma^2(G(x)) + \epsilon}, \epsilon > 0$$

where μ and σ^2 are the mean and variance of their respective inputs. The use of some positive non-zero ϵ is simply to prevent divide by 0 issues.

Furthermore, an additional similar soft constraint is required to encourage the gate to select various experts across an entire batch. This is done by ensuring that the inverse coefficient of variance of each individual expert throughout a batch is as small as possible:

$$CV'_i(x) := \frac{\mu(\sum G(x)_i)}{\sigma^2(\sum G(x)_i) + \epsilon}, \epsilon > 0$$

where $\sum G(x)_i$ represents the sum of gate weights for expert i across all input samples within a batch.

These auxiliary losses differ from those used in previous works as they do not specifically encourage any form of load balancing. As mentioned previously, this is beneficial to give the gate access to more task decompositions that do not require all tokens to be distributed equally that may be better suited for the given task.

3.3 Testing, Verification and Validation

Visdom was used to collect and analyze all data from tests and experiments in real time. The metrics used during pretraining was the reconstruction loss from MAE and the

average coefficient of variance, $CV(x)$, of the experts across a batch given by:

$$CV_i(x) := \frac{\sigma^2(\sum G(x)_i)}{\mu(\sum G(x)_i) + \epsilon}, \epsilon > 0$$

This metric was chosen as it describes the dispersion of values from the mean which correlates with how well the gate is routing tokens as a higher value suggests that it is routing tokens to different experts frequently whereas a lower value suggests that the gate has a bias towards a particular expert. From this point, this metric will be referred to as **batch-wise variation**.

3.4 Implementation

All implementation was performed on python 3.8.10 using PyTorch 2.1.2 alongside NVIDIA's CUDA 12.1.0. All training was performed on a single 24GB NVIDIA 4090 GPU which heavily limited the size of the models that could be tested.

4 Results

The model was evaluated by pretraining and finetuning on the CIFAR-100 dataset before testing its classification score on its corresponding validation set. For the sake of comparison, the same MoE model was trained utilizing different gating strategies:

- **Expert choice** - Each expert chooses top-k tokens to process [15].
- **Token choice (Capacity factor)** - Each token chooses the top-k expert to route to. If the expert's capacity is full, then the token is dropped / skipped [12].
- **Priority routing** - Each token generates a set of weights for which expert they should route to. Each expert receives the top-k tokens based on the weightings. If an expert's capacity is full, the next highest weighted expert is chosen.

Additionally, a base dense transformer model will also be trained as a baseline. The dense transformer model will have the same recipe as the MoE models except its embed dimension is increased to match the parameter count of the MoE models.

TABLE 3
Results of finetuned models on CIFAR-100

Model	Accuracy	Batch-wise variation
Dense transformer	72.4%	-
MoE	**75.1%**	**32.3**
Expert choice	74.4%	9.9
Token choice	73.2%	10.3
Priority routing	72.8%	10.2

The above Table 3 shows the final accuracies of each model on the validation set of CIFAR-100 after pretraining and finetuning. Firstly, the proposed model far outperforms its dense counterpart whilst also coming out on top compared to other routing strategies. Not only this, but as shown by Table 4, despite being trained as a dense MoE, by distilling into a sparse MoE architecture it is able to reap the benefits of conditional computing whilst maintaining high performance.

Furthermore, as shown by the above plots in Figures 2 and 3, our model is also able to train slightly faster than its competition.

TABLE 4
Table showing compute requirements of each model

Model	Parameters	GFLOPS per Image
Dense transformer	4.8M	614M
MoE	5.1M	272M
Expert choice	5.1M	272M
Token choice	5.1M	272M
Priority routing	5.1M	272M

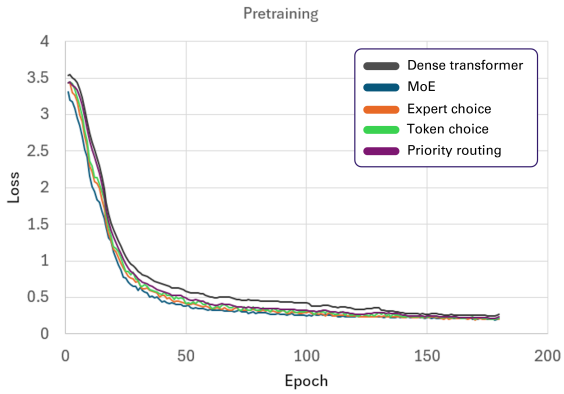

Fig. 2. Graph showing the first 175 epoch of loss for each model during pretraining.

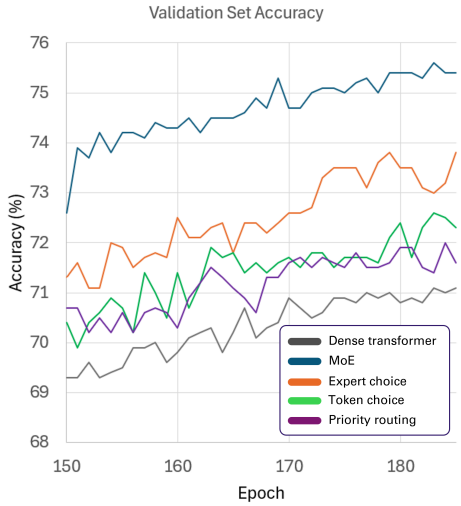

Fig. 3. Graph showing the last 200 epoch of validation set accuracy for each model during finetuning.

4.1 Investigating Expert Counts

As shown by the below Figure 4 increasing the number of experts does yield some increase in performance, however this begins to plateau at some point which matches with the findings of other papers [14] [12]. It should be noted that the point of plateau is dependent on your dataset size and

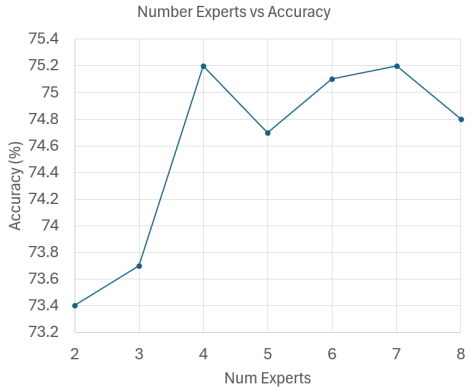

Fig. 4. Graph showing how the number of experts impacted performance of the model.

complexity of task and thus the number of experts typically is a hyperparameter that must be tuned. For CIFAR-100, it was found that this model performs best at around 4 experts and there is not much point going beyond.

4.2 Investigating Task Decomposition

Our proposed model has a much higher batch-wise variation score than the other models which suggests that it has a much more varied selection of experts across a batch of samples. This can easily be visualized by mapping the actual token distribution to each expert to see what sort of specialization each model has learned.

4.2.1 Expert choice & Token choice & Priority routing

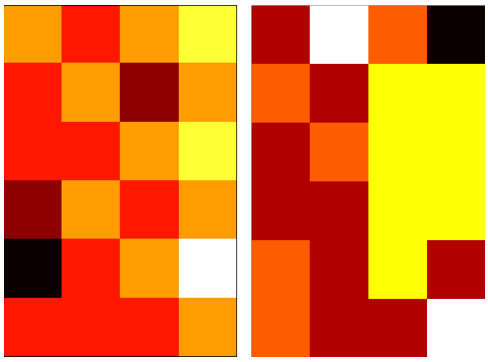

Fig. 5. Heatmap showing the average distribution of tokens across all 4 experts for each of the 6 layers for expert choice (left) and token choice (right). Priority routing is not included as it guarantees equal distribution, and so its heatmap would be uniform.

Expert choice, token choice and priority routing all exhibited the same learned routing behavior. This involved mostly unchanged token routing across all image samples which is clearly shown in the Figure 11. From this it is clear that the experts have specialized primarily in location with each expert dedicating themselves to some well-defined region.

Through the use of an esoteric image consisting of a checkerboard of 2 colors, it is clear that the experts do have some form of color bias as from Figure 12 it can be seen that different colored tokens are routed differently, however this seems to have little contribution to the overall distribution of tokens across all images. This also explains the poor batch-wise variation score from these methods as each expert routes the same tokens as a result of following a positional based task decomposition.

This further shows how routing is not as simple as trying to equally distribute compute among the experts as this inherently encourages poor task decomposition as shown by all 3 equal distributing methods converging to the same position-based expert specialization.

4.2.2 MoE

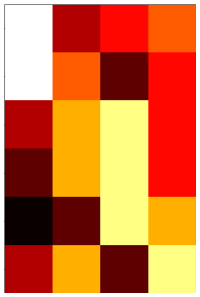

Fig. 6. Heatmap showing the average distribution of tokens across all 4 experts for each of the 6 layers.

Our proposed model, on the other hand, far outperforms the others in terms of batch-wise variation and this shows in the visualization of its token distribution in Figure 6. Overall, all experts are seemingly used across all classes, but since there is no enforcement of token balancing, there is an imbalance seen in the routing of tokens for individual groups of classes. This may seem bad, but it can actually be beneficial if the reason for the imbalance is related to the specific specialization that the model has learned.

This can be done quantitatively by inspecting the particular routes tokens take alongside their respective class. By doing so it can be shown that the imbalance is indeed caused by the specific task decomposition the gates have learned:

Fig. 7. Heatmap showing the key route taken by tokens for images under the classes "boy", "man", "woman", "girl" and "chimpanzee".

For example the above Figure 7 represents the classes "boy", "man", "woman", "girl" and "chimpanzee". This makes intuitive sense as these classes have semantic relevancy to each other. Another good example is the following patterns for household items in which the route utilized by each image very closely resembles each other further showing semantic specialization.

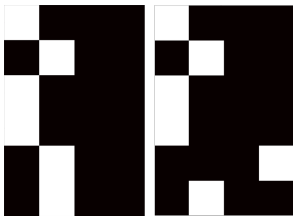

Fig. 8. Heatmap showing the key route taken by tokens for images under the classes "clock" (left) and "plate" (right).

A more obscure example would be the following 2 routes taken for the classes "clock" and "plate" as shown by Figure 8. The fact that these 2 also share similar routing shows how the model aligns with human intuition as both the clock and plate are semantically related through being large, flat circular objects.

Fig. 9. Heatmap showing the key route taken by tokens for images under the classes "lion", "wolf", "bear", "beaver", "fox", "cattle", "mouse", "skunk", "otter", "leopard", "tiger", "squirrel", "camel", "kangaroo", "elephant", "porcupine", "rabbit", "crocodile", "racoon", "possum", "shrew".

Finally, 21 of the 100 classes, all relating to 4 legged animals are found using the route found in Figure 9. This is most likely the reason for the imbalanced distribution of experts across all classes as how each class semantically relates to one another may not be balanced. To this end, as this model is able to specialize on more semantic features, its no surprise that inspecting the distribution of individual tokens also shows semantic meaning. For example Figure 10 shows a clear specialization with in the final layer with experts 1 and 3 handling the tree and ground respectively.

4.3 Additional Results

The novel technique of dropout [12] was investigated in an attempt to manage poor expert utilization through a model over utilizing a select few experts. However, using dropout seemed to cause the majority of processing to move from the experts into the attention mechanism. This was evident by comparing the accuracy of the model with all experts enabled vs with all experts dropped out. As shown by Table 5, When a model was trained without dropout its performance would plummet in response to losing its experts, but if dropout was used, then the mode would only lose a small fraction of its performance.

This is most likely because at the small scale of 4 experts, dropping out a single expert is incredibly detrimental as compared to dropping out 1 expert in a model consisting of 100+ which is how dropout was originally used in relation to MoEs. For this reason I suspect that dropout is not a feasible option for encouraging balanced expert utilization at small expert counts.

TABLE 5
Table showing compute requirements of each model

Model	Experts enabled	Accuracy
MoE	Yes	75.3%
MoE	No	24.8%
MoE (10% dropout)	Yes	74.7%
MoE (10% dropout)	No	65.6%

4.4 Ablation Study

An ablation study was performed to determine if existing MoE methods produce the optimal task decomposition for a given task. This was done by training 2 models where one used Token Choice and another used a predetermined routing algorithm. Since it is not feasible for a manual routing algorithm to be designed for a large dataset such as CIFAR-100, a smaller dataset of MNIST [22] was used.

Both models consisted of a single transformer layer with 5 experts and the model which used manual routing had the tokens routed based on similar looking classes such that expert 1 handled the digits [1, 7], expert 2 [2, 5], expert 3 [3, 4], expert 4 [6, 9] and finally expert 5 [8, 0]. The routing network was then trained after by freezing the expert weights. The results can be seen in the table below:

TABLE 6
Results of manual vs learnt task decomposition

Task Decomposition Method	Accuracy
Token Choice	88.7%
Manual	91.4%

The results above show that a manually designed task decomposition can out perform a learnt one as specific domain knowledge about the data can be used to enforce a more optimal expert specialization. This can further be shown in the visualisation of how tokens are distributed among experts by image class. Figure 13 shows that no clear intuitive expert specialization was learnt when Token Choice was used, whereas the model with manual routing has a clear expert specialization (Figure 14).

	0	1	2	3	4	5	6	7	8	9
E1	0	0	0	53	0	16	0	30	72	16
E2	0	46	0	0	54	0	74	21	33	24
E3	32	0	0	36	0	3	15	3	0	8
E4	0	32	0	22	0	82	0	10	0	32
E5	63	20	103	0	52	0	0	43	2	3

Fig. 13. Table showing the distribution of 1000 MNIST images among experts for each class of MNIST with Token Choice. The expert for a given image was determined by which expert was given the most tokens.

	0	1	2	3	4	5	6	7	8	9
E1	6	92	0	3	7	0	0	97	0	10
E2	0	0	93	12	0	92	3	0	0	4
E3	8	0	3	88	96	6	0	3	0	0
E4	0	5	0	4	4	0	85	0	0	83
E5	96	8	0	0	0	2	0	2	89	0

Fig. 14. Table showing the distribution of 1000 MNIST images among experts for each class of MNIST with Manual Routing. The expert for a given image was determined by which expert was given the most tokens.

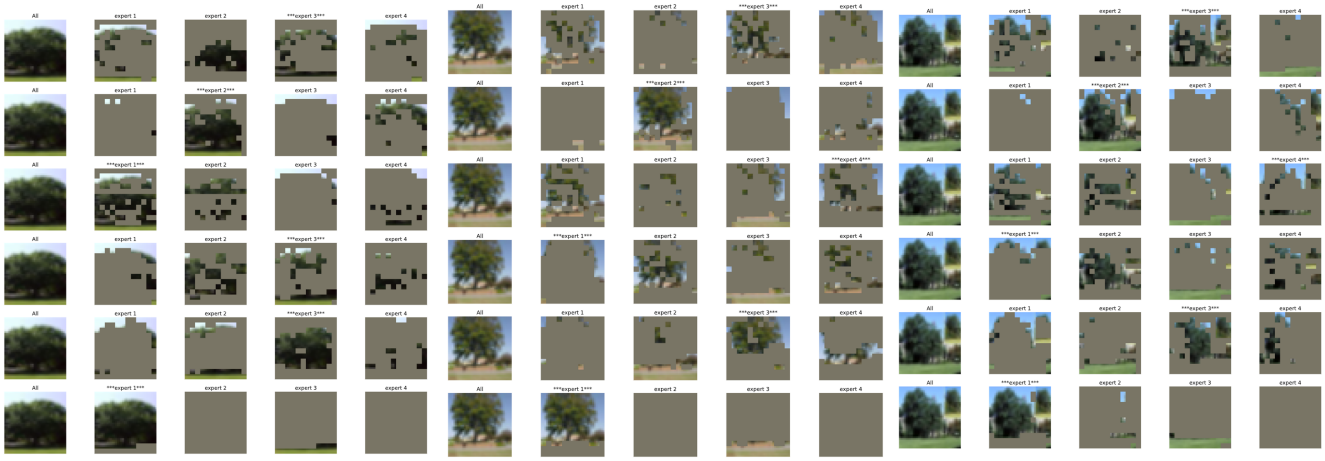

Fig. 10. Distribution of tokens for our proposed method. Notice how experts are not specialized based on location, but have more semantic specialization such as the separation of tree with the ground with experts 1 and 3 in the final layer.

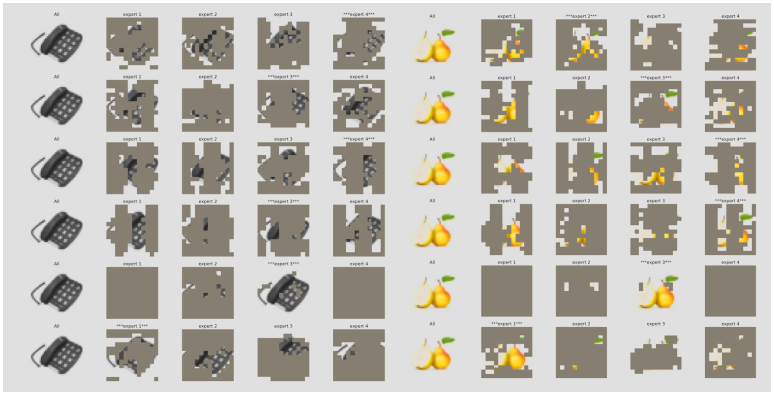

Fig. 11. Distribution of tokens for expert choice. Notice how the regions each expert looks at remains the same in positional space across both examples.

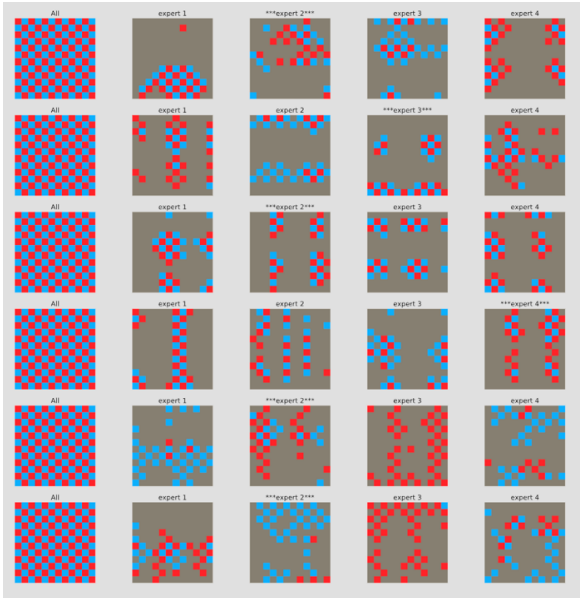

Fig. 12. Distribution of tokens for expert choice showing specialization in color. Notice how certain experts only receive tokens of a particular color.

5 Evaluation

This project will be evaluated using a list of deliverables of increasing difficulty, primarily assessing various aspects of the final model such as its ability to specialize, compute cost and its ability to overcome the shortcomings of previous models. The selected deliverables are as followed:

1) Implement a model which is able to overcome the shortcomings of sparse MoEs being non-differentiable to harness both conditional computation and fully differentiable training.
2) Implement a model which is able to decompose the task of classification into well defined specializations that ideally follow from semantics.
3) Develop a deep understanding of the model produced, and thereby the subset of MoEs surveyed, and to show the evidence for or against the various contributions made in the field.

Overall this project was extremely successful, with all deliverables being fully delivered. The success of the first deliverable is demonstrated in the core method of how the model is trained while deliverable 2 and 3 are shown extensively in Section 4.2.2.

The ablation studies and preliminary experiments conducted provided sufficient data to gain significant insight into the inner workings of the model providing evidence towards its improved task decomposition and expert specialization. The most notable results came from comparing the expert routes with the corresponding classification classes as they demonstrate the models semantic understanding of the images thoroughly.

The main limitations of this project were due to resource constraints and time. MoEs perform the best with large models and huge datasets, however with a single GPU setup and minimal onboard memory, it was difficult to test the ideas at a scale where they would be most applicable. The most significant limitation was with batch size, as MoEs often require large batch sizes to ensure that all experts are able to train on a sufficient amount of inputs per batch. Due to the memory limits enforced by the use of a single GPU, the batch size used during training had to be small. It is for this reason that the claims in this paper must be verified at a larger scale as it may not be quite a fair comparison as the use of a dense MoE during training circumvents this issue for our proposed method, which may have made the difference against the other sparse models which had to handle a smaller batch size than they were perhaps designed for.

6 Conclusion

This project successfully created a simple and novel expert routing algorithm which solves the issue of non-differentiability. Transfer learning is used to distill knowledge from a dense differentiable model to a sparse non-differentiable model after training to regain the benefits of conditional computing. Furthermore, a novel approach to the concept of load balancing is investigated. Here, experts are not be forced into equally distributing tokens; this encourages poor task decomposition and should rather be evaluated based on the variation in expert selection across a batch of different inputs.

The main novel findings of this project are as follows, in order of significance:

- Equally distributing tokens among experts is not the most optimal way to train MoEs and encourage specialization. In fact, it can be detrimental to the model as it enforces a poor set of task decompositions.
- Models trained with a softer requirement for equal load balancing are able to create specializations that align with human intuition between semantically similar images.
- Dropping out experts at a small scale significantly harms the performance of small scale MoE models as each expert has a much larger contribution overall. This often causes the model to make up for this by becoming more robust through the attention mechanism.
- Inspecting the route of tokens within a MoE model and what outputs they relate to can help with understanding how the model works under the hood.

Future extensions to this project would involve tuning the auxiliary loss terms to account for the distribution of data in a dataset. This could aid in encouraging less obvious specializations within already similar groups of inputs. For example, Figure 9 shows how this single route handles 20% of the classes in CIFAR-100, and although they are semantically related, it would be better if this individual group was further divided to utilize more of the experts overall.

REFERENCES

[1] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[2] M. Hearst, S. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications*, vol. 13, no. 4, pp. 18–28, 1998.

[3] L. Breiman, "Random forests," vol. 45, no. 1, pp. 5–32.

[4] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information Processing Systems*, D. Touretzky, Ed., vol. 2. Morgan-Kaufmann, 1989.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[6] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

[7] K. He, X. Chen, S. Xie, Y. Li, P. Dollar, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings - 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*, ser. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, 2022, pp. 15 979–15 988.

[8] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, "Mistral 7b," 2023.

[9] E. Bengio, P.-L. Bacon, J. Pineau, and D. Precup, "Conditional computation in neural networks for faster models," 2016.

[10] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[11] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. V. Le, G. E. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[12] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: scaling to trillion parameter models with simple and efficient sparsity," *J. Mach. Learn. Res.*, vol. 23, no. 1, jan 2022.

[13] D. Lepikhin, H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer, and Z. Chen, "Gshard: Scaling giant models with conditional computation and automatic sharding," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[14] C. Riquelme, J. Puigcerver, B. Mustafa, M. Neumann, R. Jenatton, A. Susano Pinto, D. Keysers, and N. Houlsby, "Scaling vision with sparse mixture of experts," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 8583–8595.

[15] Y. Zhou, T. Lei, H. Liu, N. Du, Y. Huang, V. Zhao, A. M. Dai, z. Chen, Q. V. Le, and J. Laudon, "Mixture-of-experts with expert choice routing," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 7103–7114.

[16] T. Liu, M. Blondel, C. Riquelme, and J. Puigcerver, "Routers in vision mixture of experts: An empirical study," 2024.

[17] R. Csordás, P. Piękos, K. Irie, and J. Schmidhuber, "Switchhead: Accelerating transformers with mixture-of-experts attention," 2023.

[18] M. I. Jordan and R. A. Jacobs, "Hierarchical mixtures of experts and the em algorithm," *Neural Comput.*, vol. 6, no. 2, pp. 181–214, mar 1994.

[19] J. Puigcerver, C. Riquelme, B. Mustafa, and N. Houlsby, "From sparse to soft mixtures of experts," 2023.

[20] J. H. Tan, "Pre-training of lightweight vision transformers on small datasets with minimally scaled images," 2024.

[21] E. D. Cubuk, B. Zoph, D. Mané, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 113–123.

[22] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.