Towards Interpretability and Transferability in Mixture of Experts

Yamuna Krishnamurthy

ROYAL HOLLOWAY UNIVERSITY OF LONDON

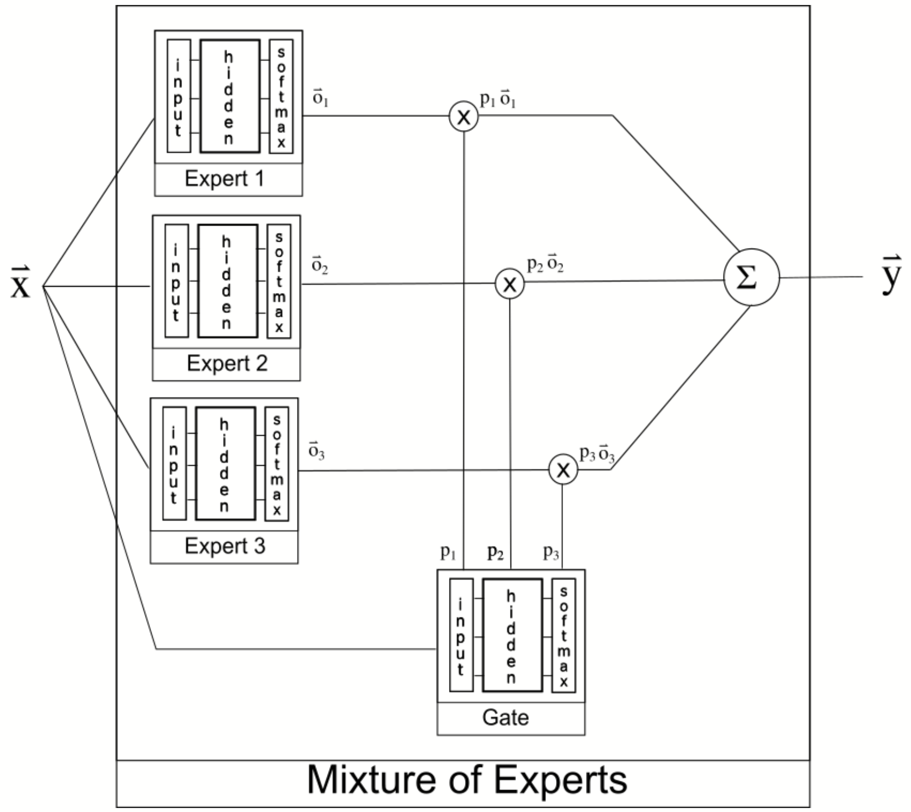
Talk Outline

- Why modular neural networks (MNN)?
- Overview of Mixture of Experts (MoE), a gated modular neural network architecture
- What is interpretability with MoE?
- Do existing MoE architectures learn an interpretable task decomposition?
- Why do existing MoE architectures not learn an interpretable task decomposition?
- Proposed solutions for better MoE task decomposition for interpretability and transferability

Motivation

- Modular neural networks (MNN) have recently been demonstrated to achieve high performance with much less parameters Shazeer et al [1], Rajbhandari et al [2]
- They reduce inference time by conditional computation
- MNN could be inherently more interpretable
  - Potentially learn insightful problem decomposition
  - Modular attribution of errors
- Can facilitate modular transferability
- Modules could be multi-modal Kaiser et al [3]
- Potential for continual learning Hihn et al [4]

Mixture of Experts (MoE) architecture



- Gated modular neural network
- Set of simple independent neural networks called "experts"
- Simple neural network that works as a gate or soft-switch
- Gate learns the sample based expert selection policy
- Gate allocates samples to experts, decomposing the task between experts
- Model output is some combination or selection of the outputs of the individual experts
- Experts and gate are trained together end-to-end

MoE Architectures

- Fascinatingly, many MoE architectures can be realized with different:
  - inference methods (eg. expected sum of or expert outputs, stochastic selection of expert outputs)
  - loss computations (eg. expected sum of expert losses)
  - regularizations (eg. expert importance regularization), and
  - expert and gate training methods (eg. SGD, EM)

MoE Architectures : Stochastic Model

1. **Stochastic Model** [5]:

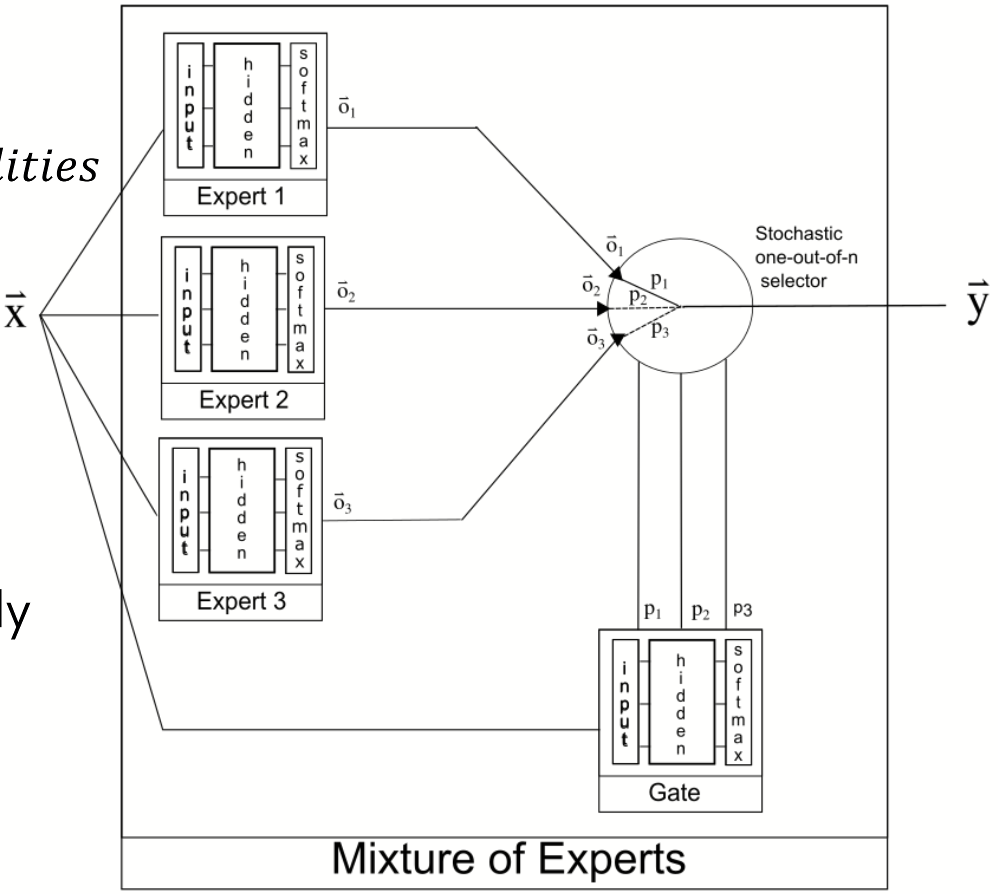
M *is the number of experts*

$\vec{p} = \{p_1, p_2, \dots, p_M\}$ *are the gate probabilities*

$$\vec{y} = \overrightarrow{o_k}, \qquad k = argmax_{i \in M}(p_i)$$

$$L_{\vec{x}} = \sum_{i=1}^{M} p_i \cdot loss(d, \overrightarrow{o_i})$$

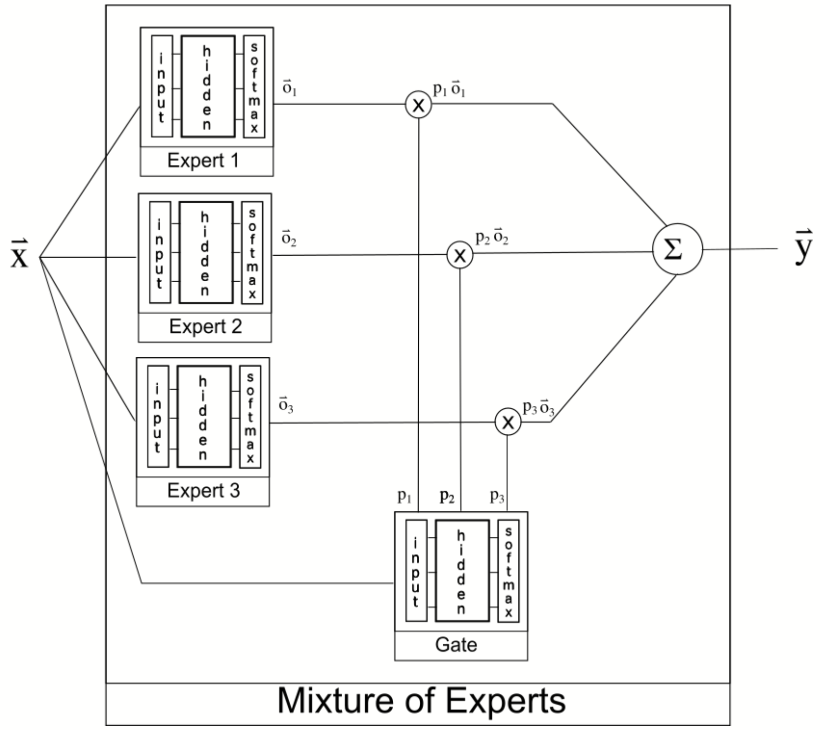
- Stochastic model allows conditional computation during inference as only one expert is chosen.
- Inference and training are different
- Experts are truly independent



MoE Architectures : Expectation and Pre-Softmax Models

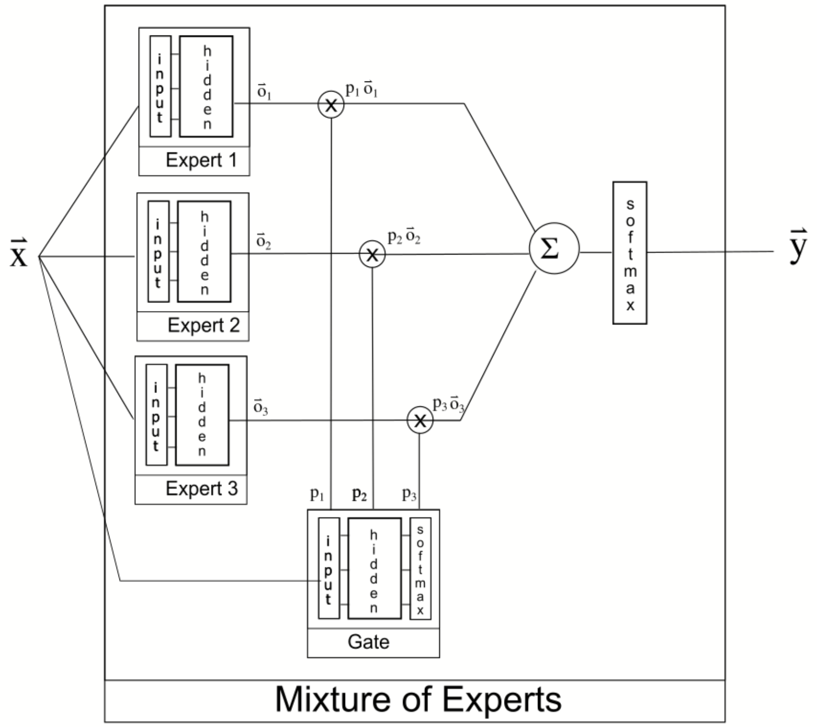
2. Expectation Model [6]:

$$\vec{y} = \sum_{i=1}^{M} p_i \, \overrightarrow{o_i}$$
$$L_{\vec{x}} = loss(d, \vec{y})$$



3. Pre-softmax Model :

$$\vec{y} = softmax\left(\sum_{i=1}^{M} p_i \, \overrightarrow{o_i}\right)$$
$$L_{\vec{x}} = loss(d, \vec{y})$$

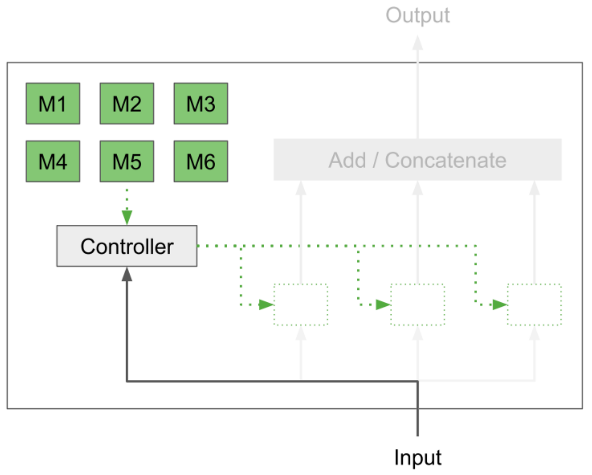


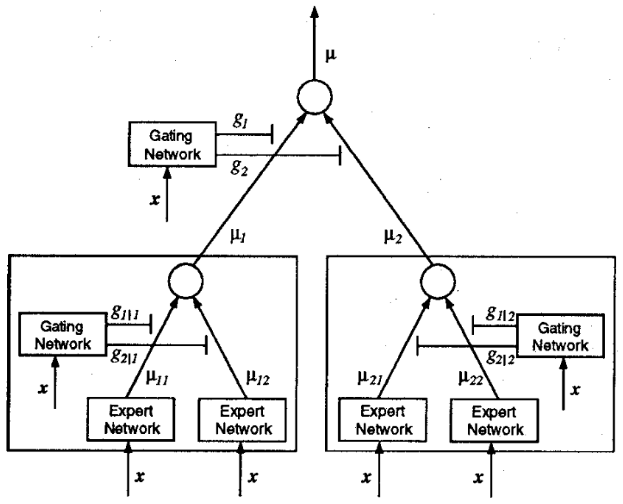
MoE Architectures : Expectation and Pre-Softmax Models

- Output is a linear combination of the expert outputs and hence it is not identifiably unique
- Inference and training are the same
- Residual errors could force all or most experts to update
- Experts are coupled and hence co-operate rather than compete

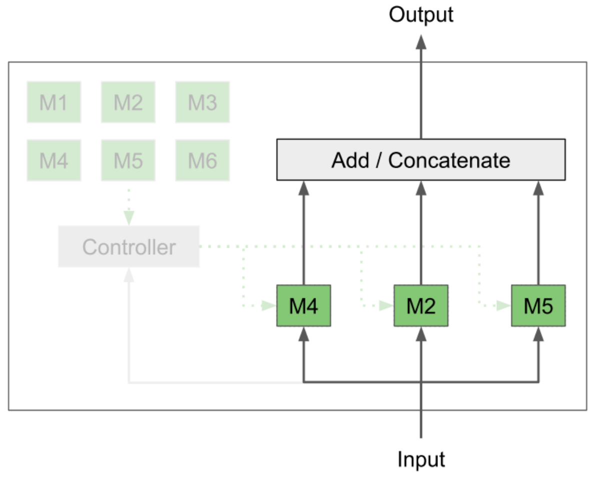
MoE Architectures: Hierarchical Models

- Jacobs et al [7] and Kirsch et al [8] designed multi-level experts and gate as generalized linear models.
- Learning is treated as a maximum likelihood problem and parameters are adjusted using Expectation-Maximization (EM) algorithm.
- [7] has experts only in the first level. [8] is a composition of experts at each level determined by the gate at that level.
- Much slower than training with SGD.





Hierarchical model from [7]



Hierarchical model from [8]

Research Questions

1. How are tasks decomposed between experts with the existing methods?
2. Does the decomposition allow interpretability by attribution?
3. What is a good problem decomposition for interpretability?
4. Does the task decomposition among the experts enable transferring them to other tasks?
5. How to train MoE for better interpretability and transferability?
6. Does the same decomposition of the tasks enable both interpretability and transferability? Or do they need different task decompositions?

What is interpretability?

- Gating network learns a meaningful decomposition of the input space into regions with natural ’rules’. For example, for a classification task it could use different modules to predict different classes; and/or
- Each module learns non-intersecting functions or subsets of the task
- Facilitates:
    - Attributing errors to the gate or
    - Attributing errors to the modules
    - Model debugging

Does the gate learn a 'natural' task decomposition?

MoE has two common pathological problems

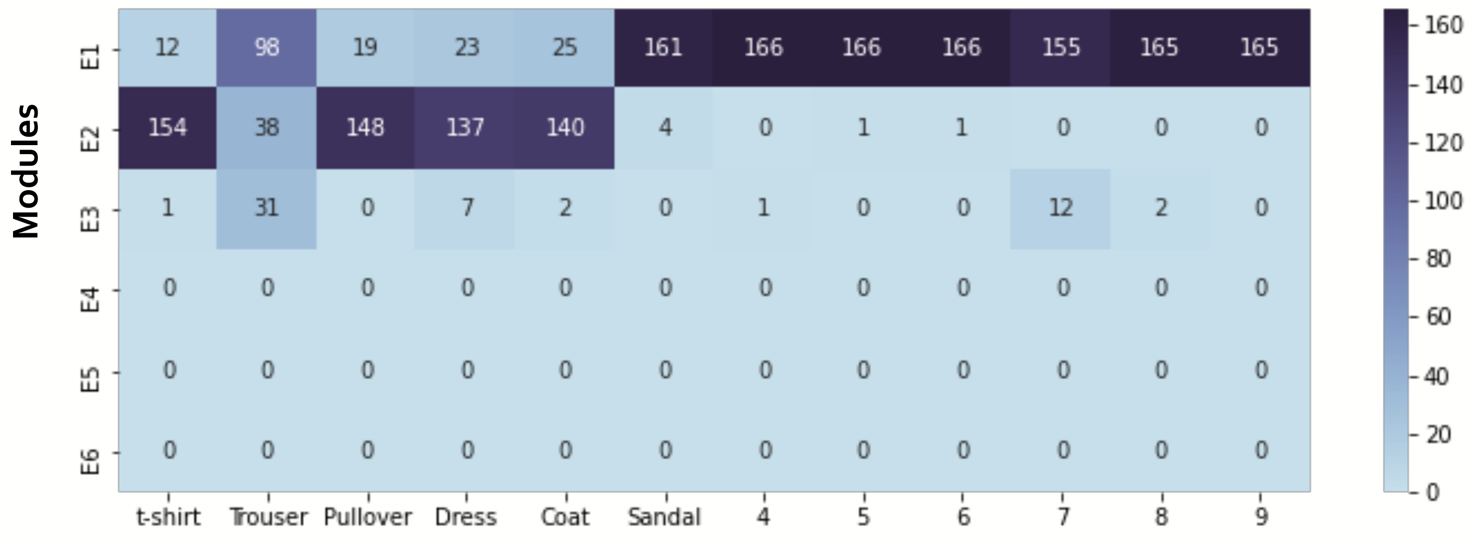
Trivial Gate: One expert learns all the classes. Known as *module collapse*, it occurs when the gate selects the same expert for all the samples. The MoE output does not depend on the gate and is in effect the same as a single model resulting in poor interpretability.

Trivial Experts: One expert learns only one class. It occurs when an expert classifies all inputs as the same class. In this case the gate does all the classification and could result in poor transferability.

- Both cases indicate poor allocation of data samples to the experts, by the gate, subsequently leading to either overuse or starvation of experts.

Does the gate learn a ‘natural’ task decomposition?

- Gate decomposition of tasks:
  - does not always guarantee interpretability
  - same module E1 is used for FMNIST and MNIST data
  - under utilizes experts



Modules	t-shirt	Trouser	Pullover	Dress	Coat	Sandal	4	5	6	7	8	9
E1	12	98	19	23	25	161	166	166	166	155	165	165
E2	154	38	148	137	140	4	0	1	1	0	0	0
E3	1	31	0	7	2	0	1	0	0	12	2	0
E4	0	0	0	0	0	0	0	0	0	0	0	0
E5	0	0	0	0	0	0	0	0	0	0	0	0
E6	0	0	0	0	0	0	0	0	0	0	0	0

Gate allocation of tasks to experts for the combined FashionMNIST and MNIST datasets

Why does the gate not learn a ‘natural’ task decomposition?

Optimality problem?

- Is there is a training disadvantage for the interpretable task decomposition, that is, the training converges slower?
- Does an interpretable task decomposition lead to a higher overall error rate?

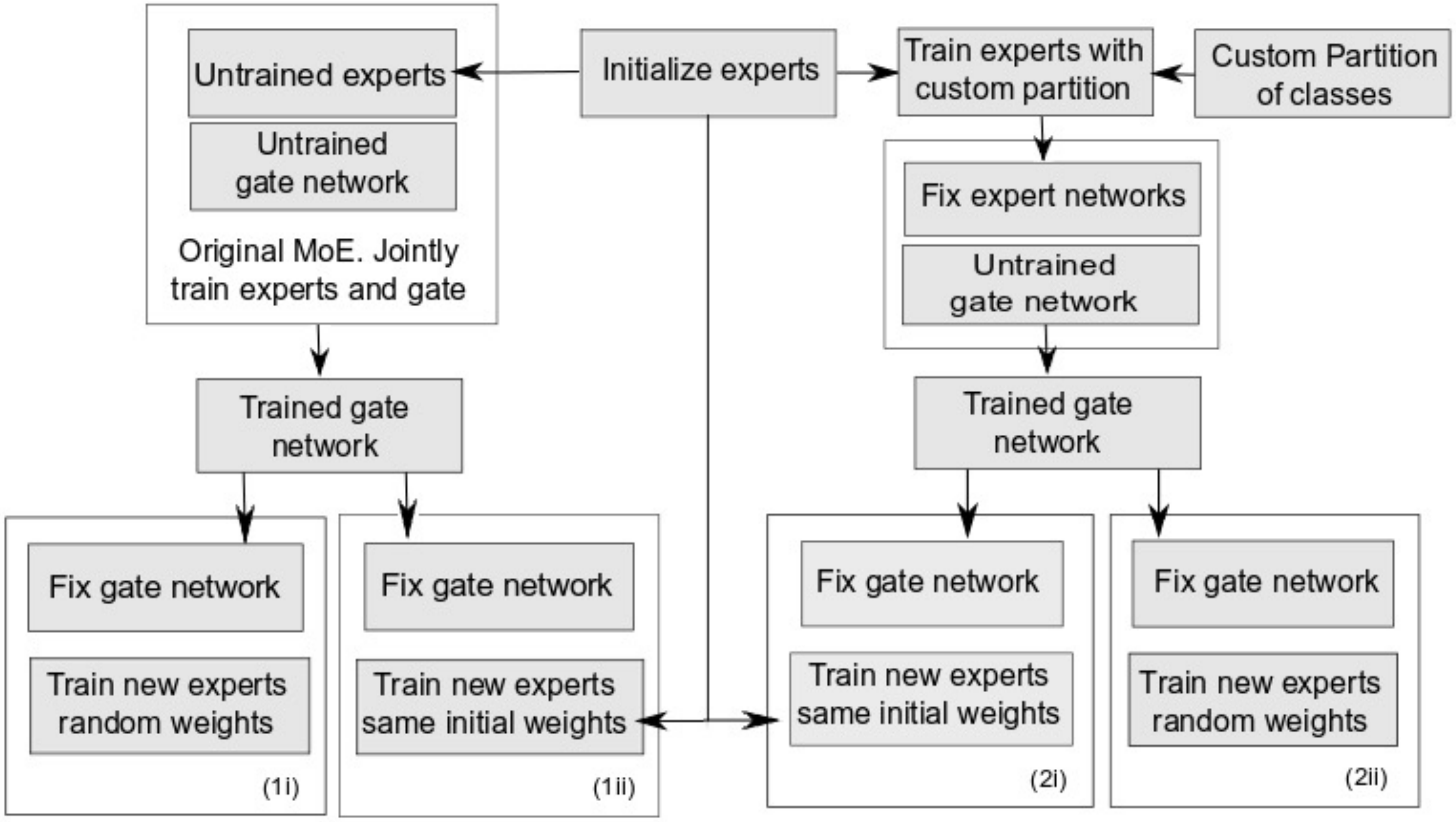
Training Problem?

- Is the heuristic of jointly optimising gate and experts sufficiently good to find a natural decomposition?

Does an interpretable task decomposition exist and can the gate learn it?

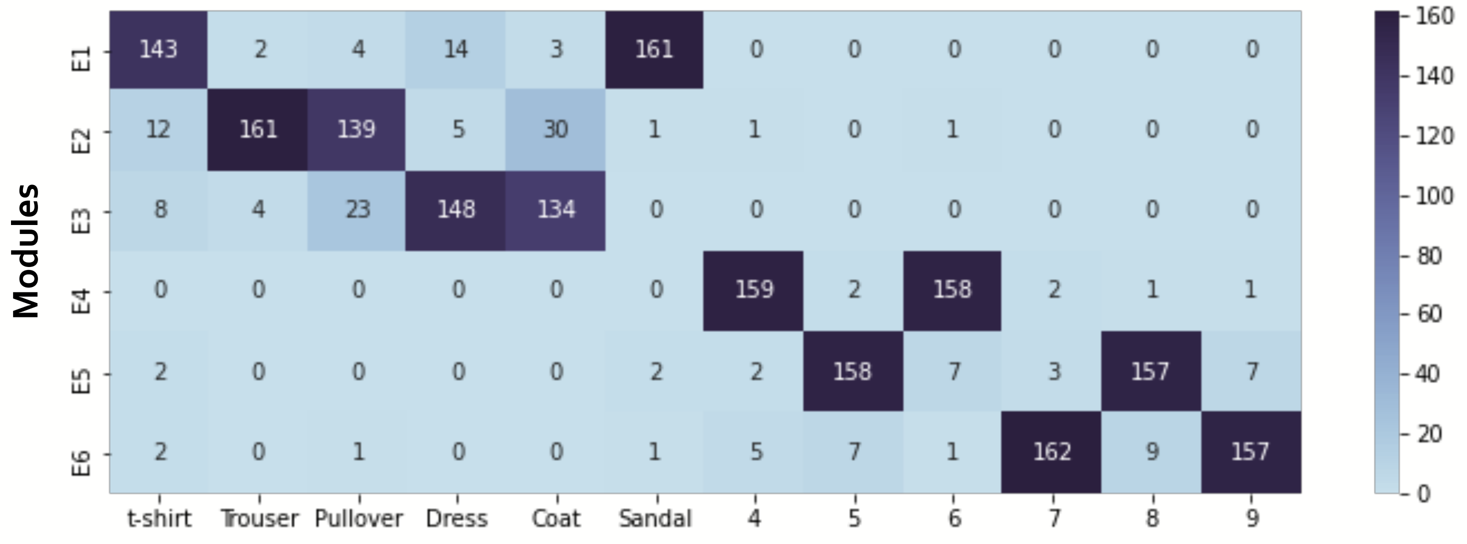
ROYAL HOLLOWAY UNIVERSITY OF LONDON

We designed an experiment to test if a gate can learn a good decomposition and if there are any disadvantages to learning it.


Untrained experts
Untrained gate network
Original MoE. Jointly train experts and gate
Initialize experts
Train experts with custom partition
Custom Partition of classes
Fix expert networks
Untrained gate network
Trained gate network
Trained gate network
Fix gate network
Train new experts random weights
(1i)
Fix gate network
Train new experts same initial weights
(1ii)
Fix gate network
Train new experts same initial weights
(2i)
Fix gate network
Train new experts random weights
(2ii)

Does an interpretable task decomposition exist and can the gate learn it?

- Pre-train experts with an interpretable split of tasks
  - for example, [t-shirt, sandal], [trouser, pullover], [dress, coat], [4,6], [5,8], [7,9]
- Gate trained on pre-trained experts decomposes the task cleanly as shown below.

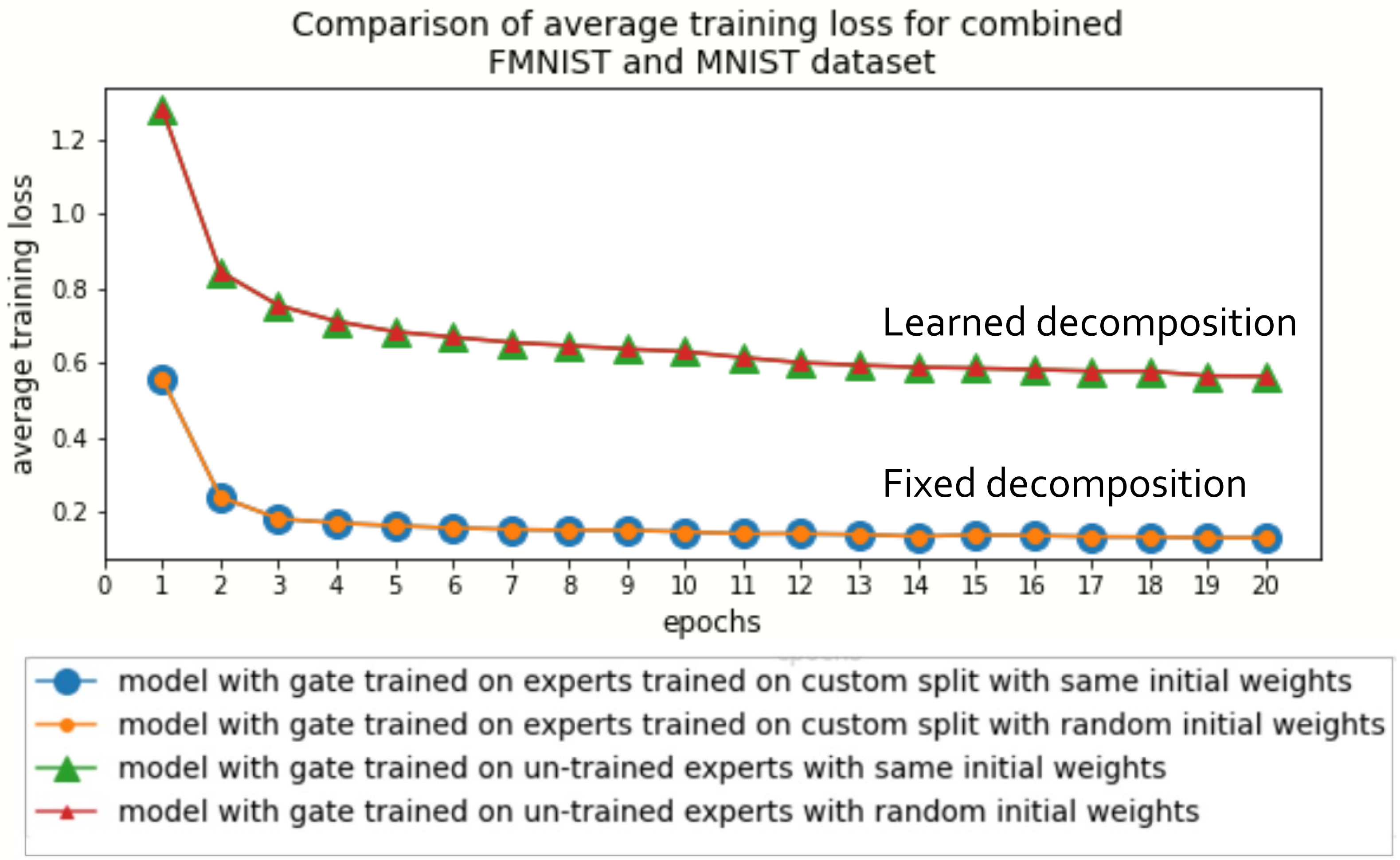


	t-shirt	Trouser	Pullover	Dress	Coat	Sandal	4	5	6	7	8	9
E1	143	2	4	14	3	161	0	0	0	0	0	0
E2	12	161	139	5	30	1	1	0	1	0	0	0
E3	8	4	23	148	134	0	0	0	0	0	0	0
E4	0	0	0	0	0	0	159	2	158	2	1	1
E5	2	0	0	0	0	2	2	158	7	3	157	7
E6	2	0	1	0	0	1	5	7	1	162	9	157

Gate allocation of tasks to experts for the combined FashionMNIST and MNIST datasets with gate trained on pre-trained modules

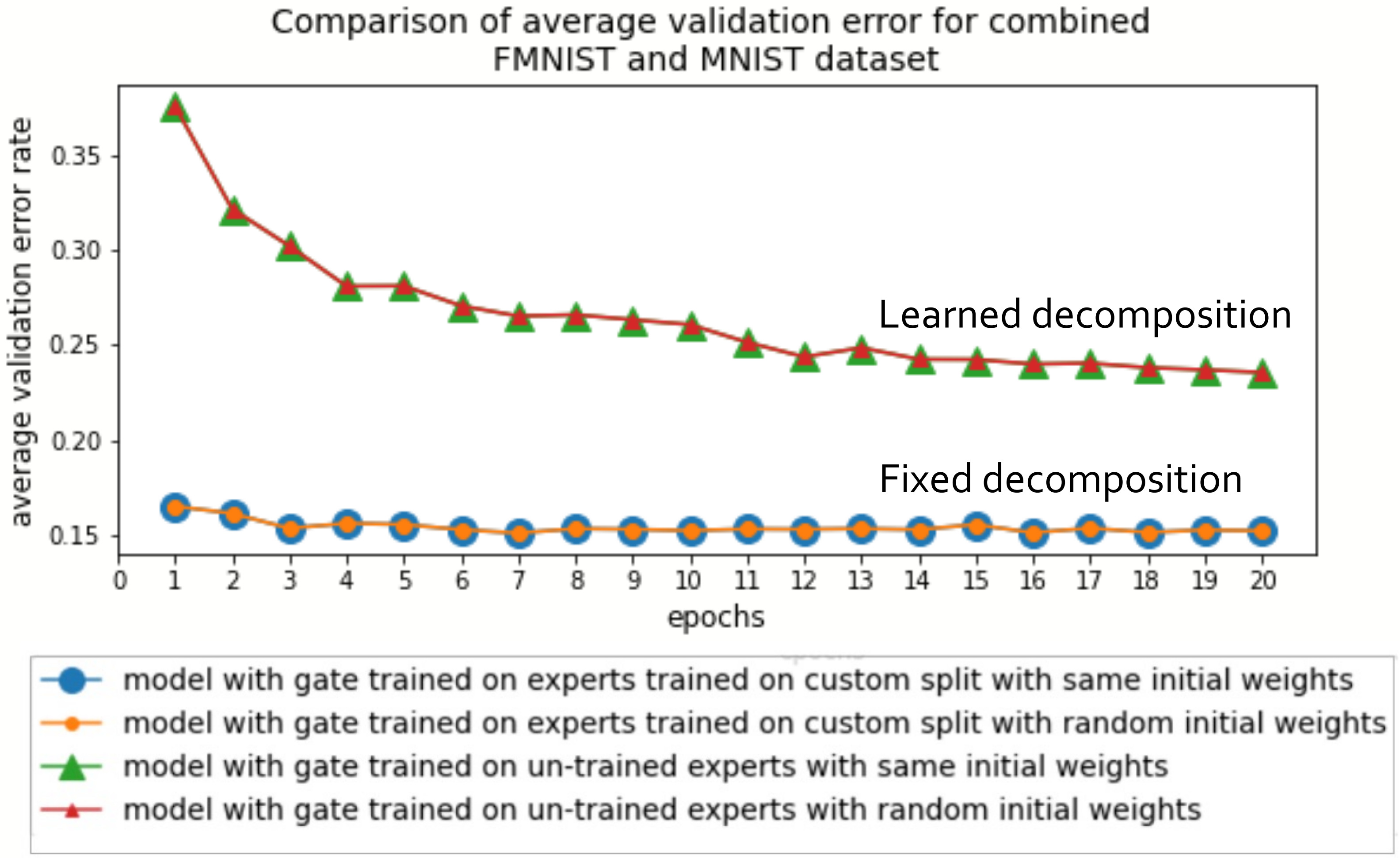
Why does the gate not learn a 'natural' task decomposition?

- Is there a learning advantage for a bad decomposition?



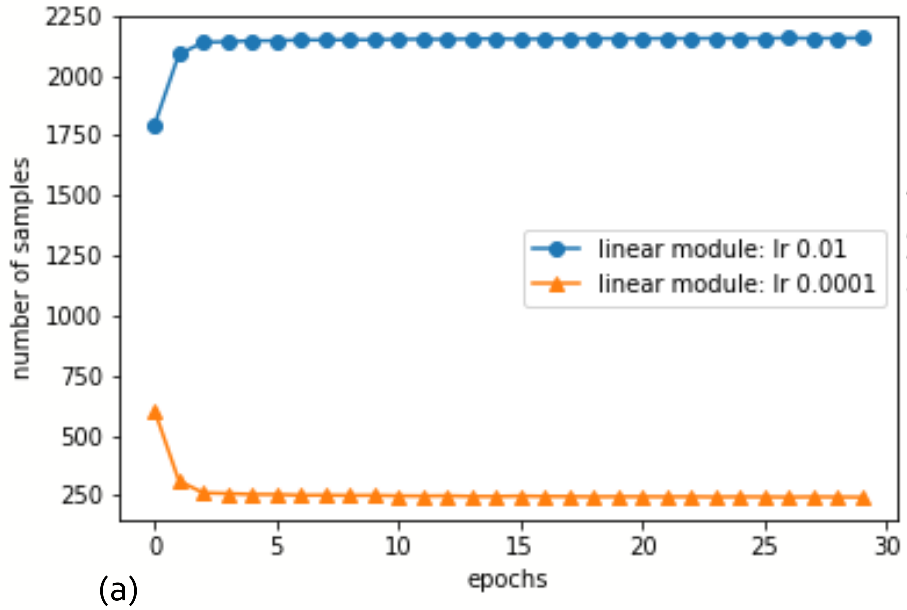
Why does the gate not learn a 'natural' task decomposition?

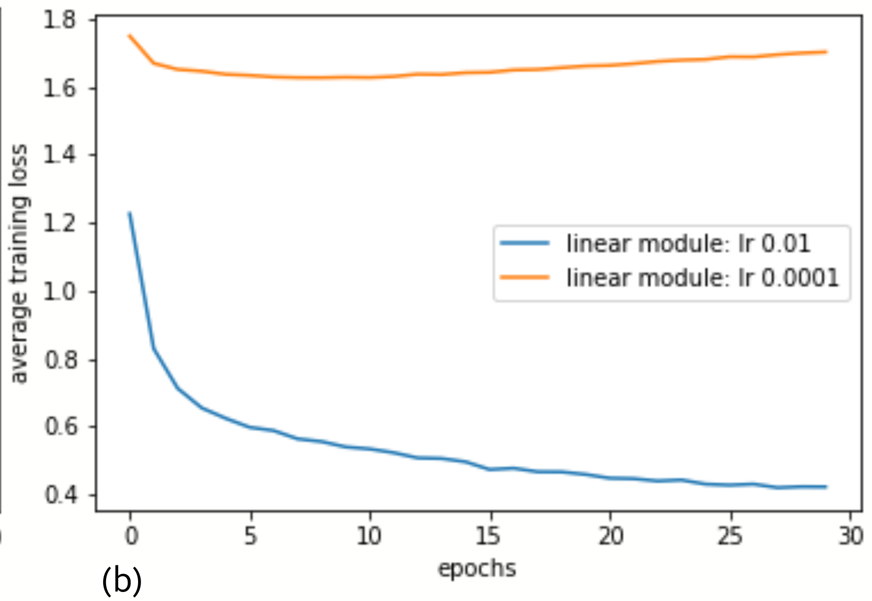
- Is there an error advantage for a bad decomposition?



Why does the gate not learn a 'natural' task decomposition?

- Is it the heuristic of jointly training the gate and experts?
  - GMNN trained with 2 experts with different learning rates
  - Expert with higher learning rate captures most of the data samples
  - Joint expert and gate training could lead to poor allocation of data samples and hence bad decompositions





(a) Sample distribution between 2 modules (b) Average training loss

For the results discussed thus far and further details on them please refer to our XAI workshop paper at NeurIPS 2021:

Yamuna Krishnamurthy and Chris Watkins. Interpretability in Gated Modular Neural Networks. *In Explainable AI approaches for debugging and diagnosis (XAI) Workshop at NeurIPS, 2021.*

Problems with 'end-to-end' MoE training and proposed solutions

Poor initial gating decisions lead to two issues that significantly affect task decomposition of MoE:

1. Poor allocation of training samples leads to poor expert utilization and starvation, and thereby to
2. Unnatural task decompositions

Proposed solutions are to improve:

1. Batch distribution to experts while training, and
2. Expert utilization

Measures of expert utilization and sample distribution?

- **Per sample gating entropy $\boldsymbol{H_b}$:** Measures per sample expert allocation. Low H_b indicates less uncertainty in selecting an expert per sample.

$$H_b = \frac{1}{B}\sum_{i=1}^{B} H(\overrightarrow{p_i})$$

B is the batch size
$\vec{p}$ is the per sample gate probabilities

- **Expert utilization entropy $\boldsymbol{H_e}$:** Measures batch expert utilization with entropy of the average of the gate probabilities over the batch. High H_e indicates a good utilization of experts.

$$H_e = H\left(\frac{1}{B}\sum_{i=1}^{B} \overrightarrow{p_i}\right)$$

- Both these measures were introduced in Kirsch et al [6]

Measures of expert utilization and sample distribution?

- **Mutual Information $I(E;Y)$ of MoE model output (Y) and expert (E):**
  - $E \in \{E_1, \dots, E_M\}$ and $Y \in \{Y_1, \dots, Y_K\}$ are the expert selection and model output random variables respectively
  - $I(E;Y)$ is a measure of the dependence of Y on E. Higher $I(E;Y)$ indicates better expert utilization
  - $I(E;Y) \equiv H(E) + H(Y) - H(E,Y)$
  - We first compute the joint probabilities $P(E,Y)$ in Table 2 from observations in Table 1
  - We then compute marginal probabilities $P(E)$and $P(Y)$ from the joint probabilities
  - We then compute the entropies from these quantities to compute mutual information

Table 1 Matrix of count of number of times E_m is selected for task Y_k

Count (E,Y)	Y_1	…	Y_K
E_1	c_{11}	…	c_{1K}
$\vdots$	$\vdots$	$\ddots$	$\vdots$
E_M	c_{M1}	…	c_{MK}

Table 2 Joint and marginal probabilities of E and Y

P(E,Y)	Y_1	…	Y_K	$P(E_m)$
E_1	$p(E_1, Y_1)$	…	$p(E_1, Y_K)$	$P(E_1)$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
E_M	$p(E_M, Y_1)$	…	$p(E_M, Y_K)$	$P(E_M)$
$P(Y_k)$	$p(Y_1)$		$p(Y_K)$	1

Proposed Solutions for Better Batch Distribution and Expert Utilization

- Existing approach
  - Batchwise expert importance regularization
- Our approach
  - Decoupling training of experts and gating
  - Attentive gating
  - Gating with sample-similarity regularization

SOTA: Expert Importance Regularization for Better Expert Utilization

- In their paper: 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*. 2017
- Jeff Dean et al at Google proposed a soft constraint approach, to enforce equal importance to all experts for a batch X, by adding a regularization term to the loss.
- Relative importance of an expert to the batch is measured as:

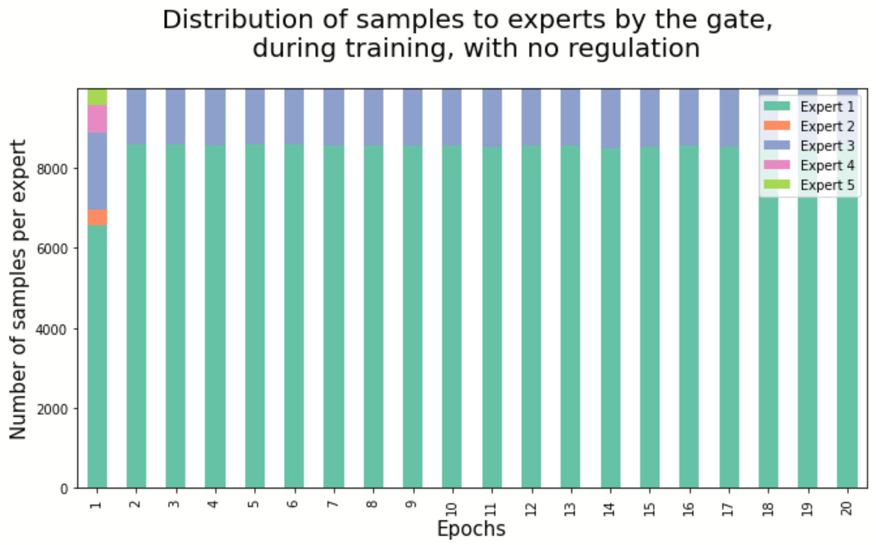
$$\vec{I} = Importance(X) = \sum_{x \in X} \overrightarrow{p_x}$$

- Goal is to assign equal importance to all experts for a batch by ensuring low coefficient of variation (CV) of $\vec{I}$ by minimizing the loss term $L_{importance}$:

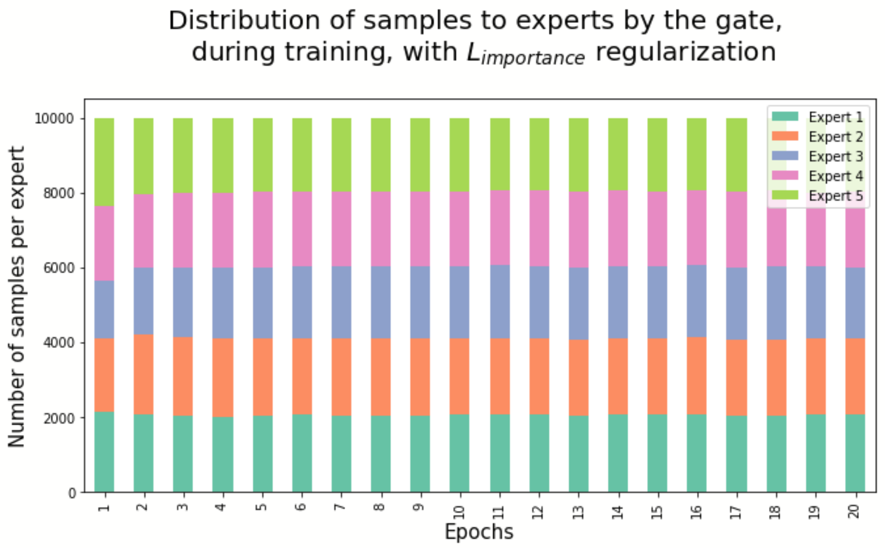
$$L_{importance}(X) = w_{importance} \cdot CV\left(\vec{I}\right)^2$$

- $w_{importance}$ is a hand-crafted parameter. $CV = \sigma(\vec{I})/\mu(\vec{I})$

SOTA: Expert Importance Regularization for Better Expert Utilization



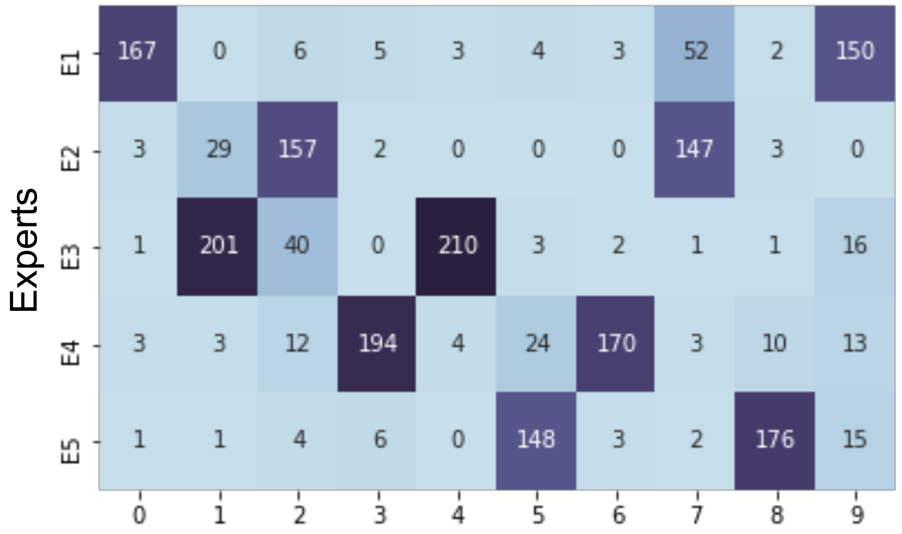
(a)



(b)

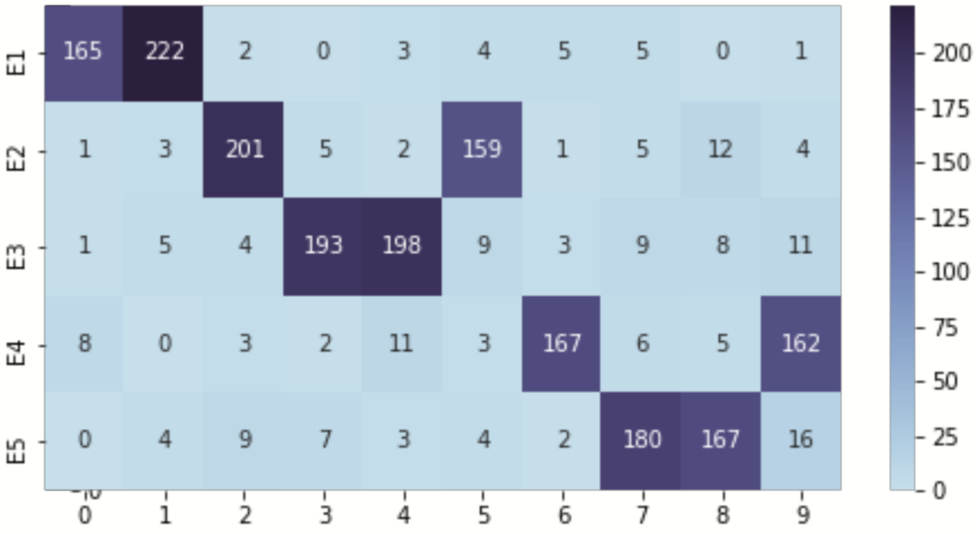
Distribution of 10,000 MNIST samples by the gate to 5 experts during training: (a) without regularization and (b) with $L\_importance$.

SOTA: Expert Importance Regularization for Better Expert Utilization



Experts	0	1	2	3	4	5	6	7	8	9
E1	167	0	6	5	3	4	3	52	2	150
E2	3	29	157	2	0	0	0	147	3	0
E3	1	201	40	0	210	3	2	1	1	16
E4	3	3	12	194	4	24	170	3	10	13
E5	1	1	4	6	0	148	3	2	176	15

(a)

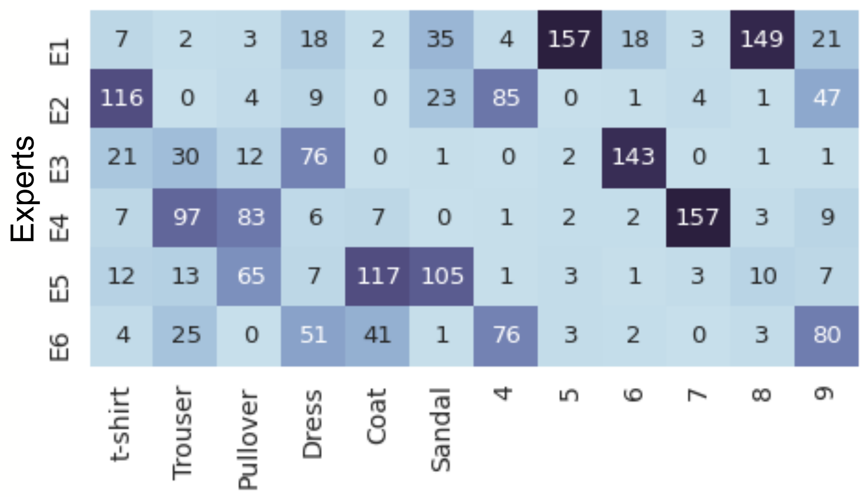


	0	1	2	3	4	5	6	7	8	9
E1	165	222	2	0	3	4	5	5	0	1
E2	1	3	201	5	2	159	1	5	12	4
E3	1	5	4	193	198	9	3	9	8	11
E4	8	0	3	2	11	3	167	6	5	162
E5	0	4	9	7	3	4	2	180	167	16

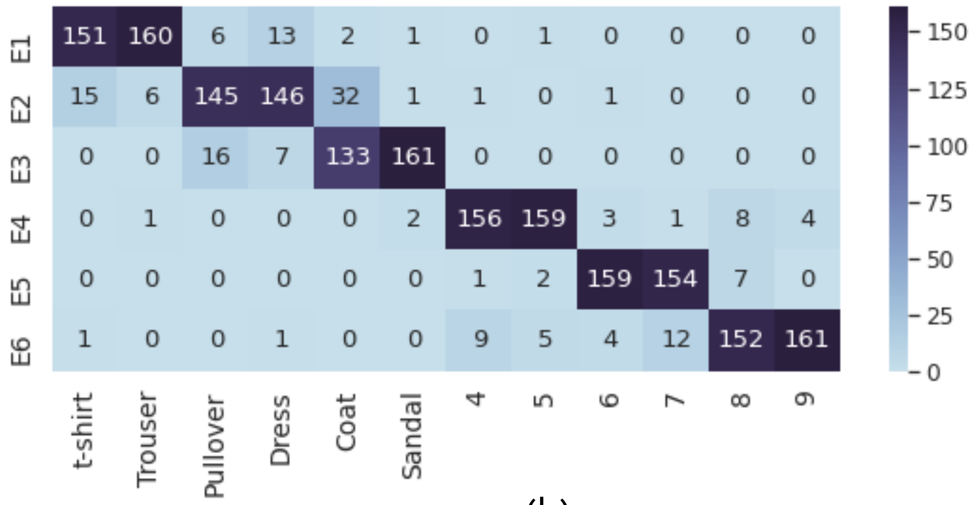
(b)

Inference on 2000 samples of MNIST test data: (a) Experts used by the gate for classification of each class with $L_{importance}$ regularization; (b) Experts used by the gate for classification of each class with pre-trained experts for custom splits of the dataset.

SOTA: Expert Importance Regularization for Better Expert Utilization



(a)



(b)

Inference on 2000 samples of combined FashionMNIST and MNIST test data: (a) Experts used by the gate for classification of each class with $L_{importance}$ regularization; (b) Experts used by the gate for classification of each class with pre-trained experts for custom splits of the dataset.

Our Approach: Decoupling training of gate and experts using dual temperature softmax

ROYAL HOLLOWAY UNIVERSITY OF LONDON

Motivation

- MoE's end-to-end joint training tightly couples the expert and gate training
- Gate's softmax output produces a hard probability distribution. It converges to high probabilities for the preferred expert and low probabilities for the other experts
- This leads to data starvation of the unselected experts resulting in module collapse
- *Decouple the gate and expert learning* by smoothing the gate's probability distribution for learning the experts but use the harder distribution to learn the gate.

Our Approach: Decoupling training of gate and experts using dual temperature softmax

Method

- Softer gate distribution can be achieved by regulating the gate's softmax with a temperature T applied to its logit input
- Soft softmax has been used by Hinton et al [9] for model distillation
- We train gate and experts on different losses

Gate loss with hard softmax:

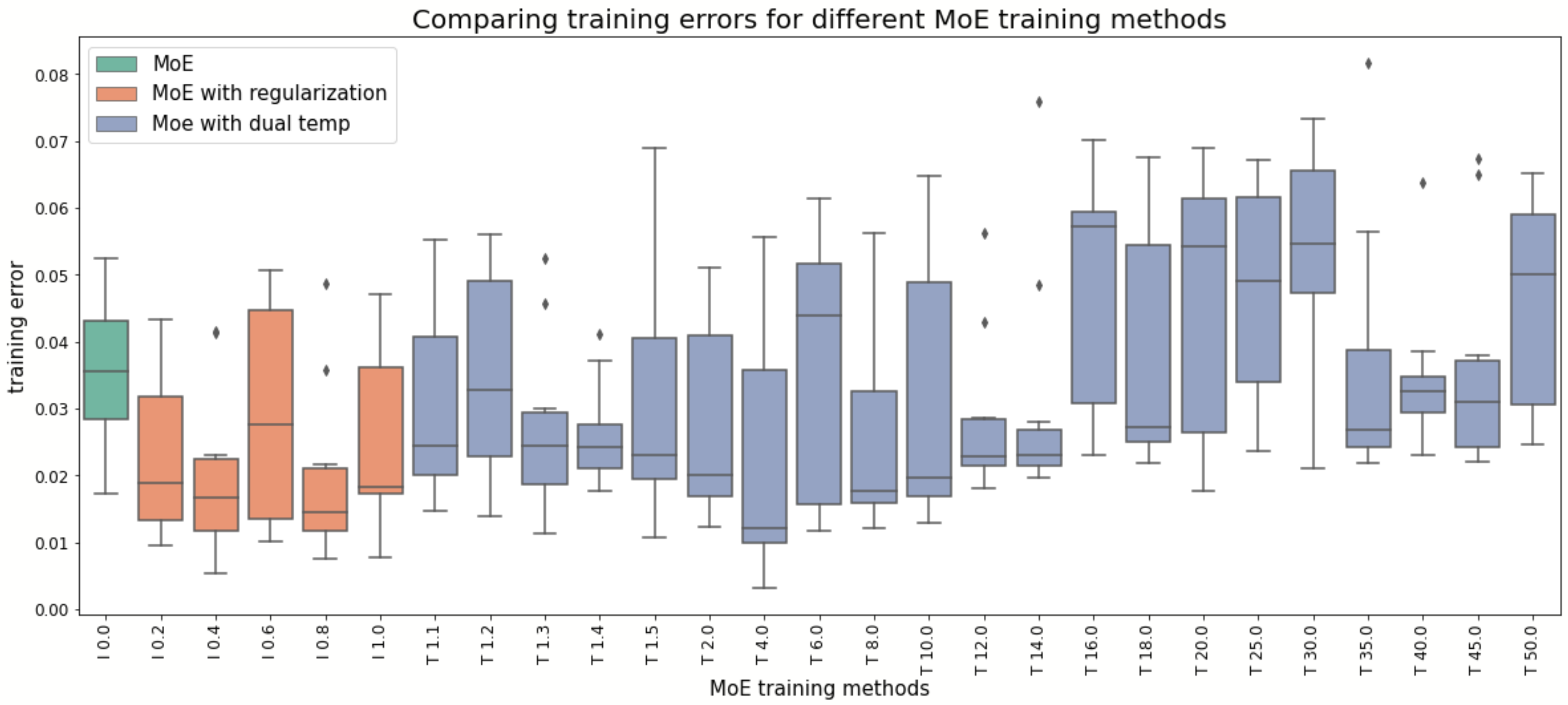
$$L_g = loss\left(d, \sum_{i=1}^{M} p_i \, \vec{o_i}\right) where \; p_i = \frac{exp(z_i)}{\sum_{j=1}^{M} exp(z_j)}$$

z_i is the logit for expert i

Expert loss with soft softmax:

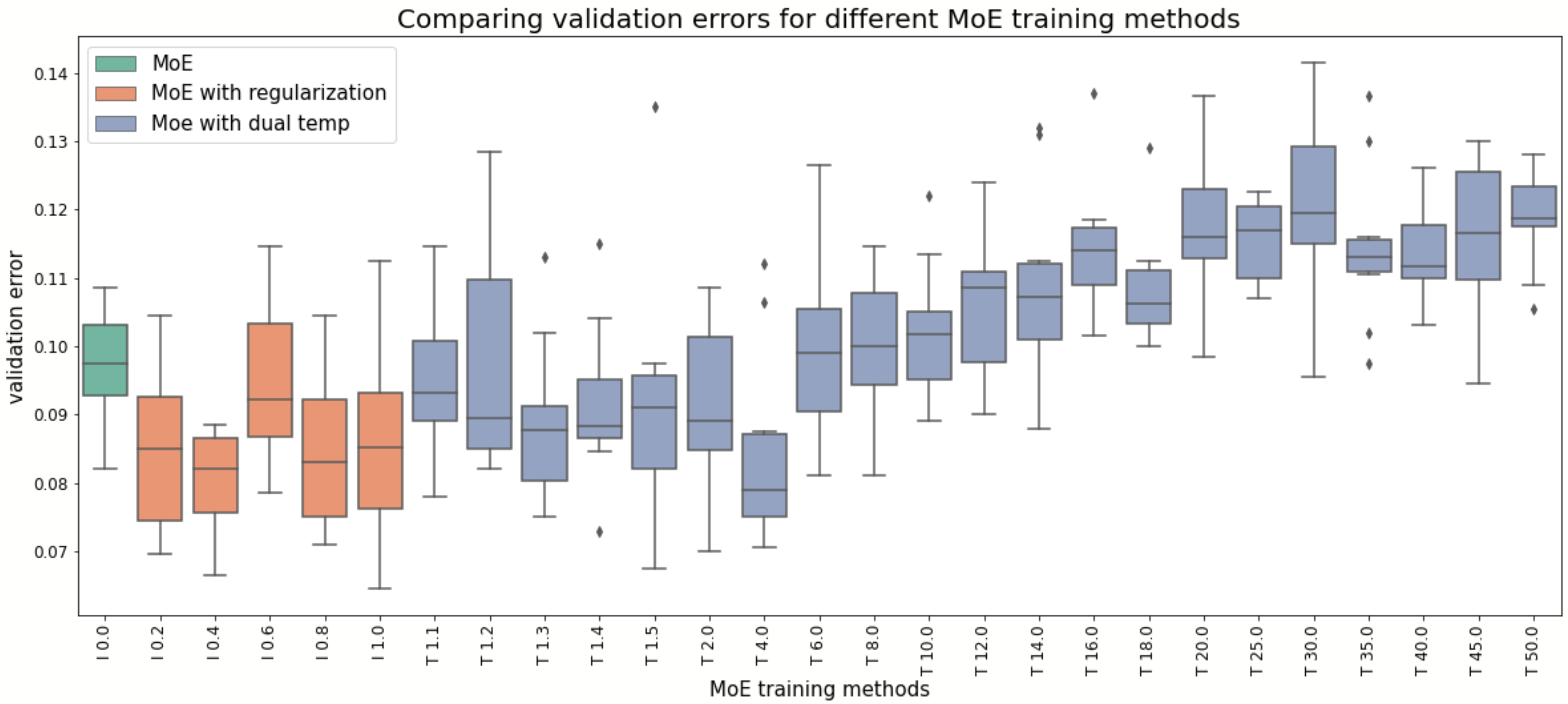
$$L_e = loss\left(d, \sum_{i=1}^{M} p_{it} \, \vec{o_i}\right) where \; p_{it} = \frac{exp(z_i/T)}{\sum_{j=1}^{M} exp(z_j/T)} \; for \; T > 1$$

Our Approach: Decoupling training of gate and experts using dual temperature softmax



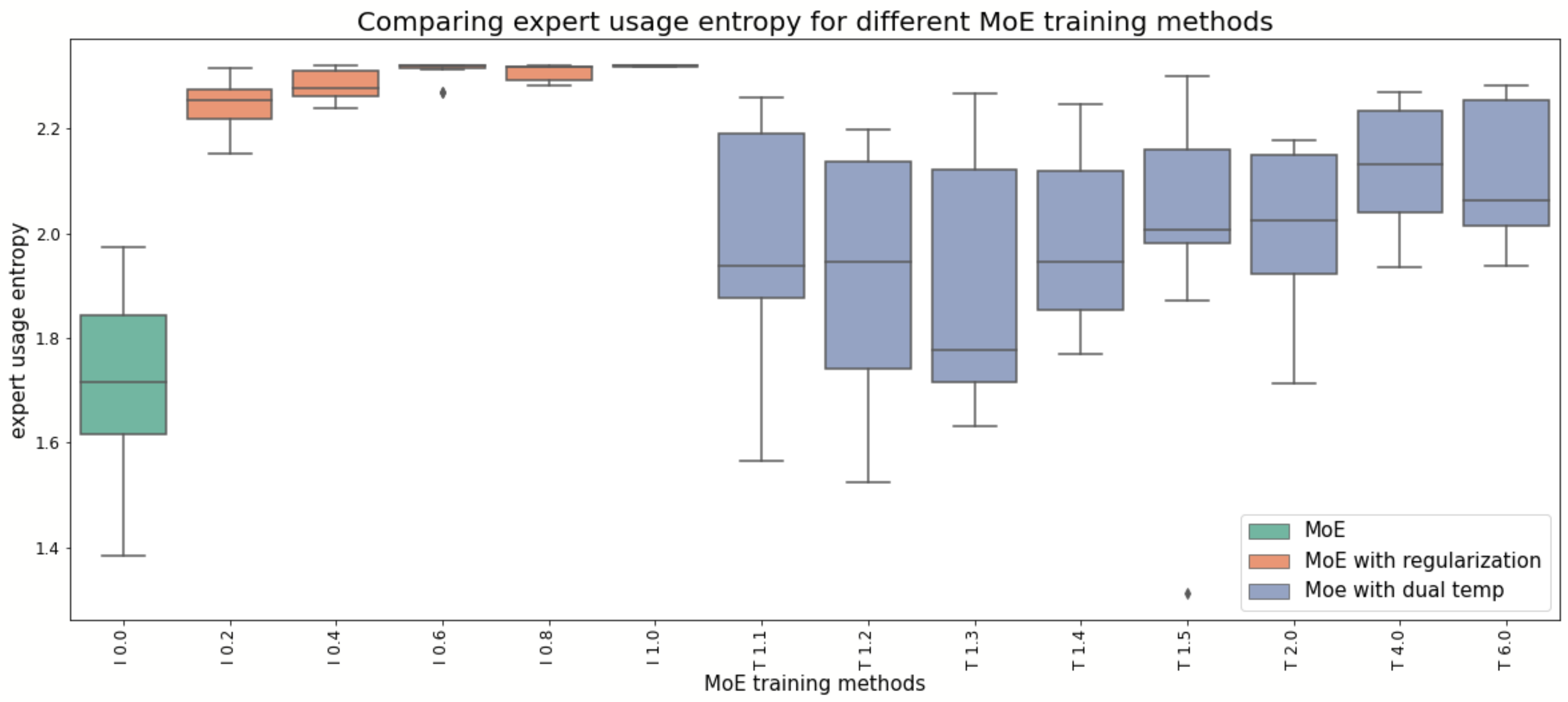
Comparing training error for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}$ $(\mathrm{I}) \in \{0.2, \dots, 1.0\}$ and dual temperature training with temperature $T \in \{1.1, \dots, 50\}$

Our Approach: Decoupling training of gate and experts using dual temperature softmax



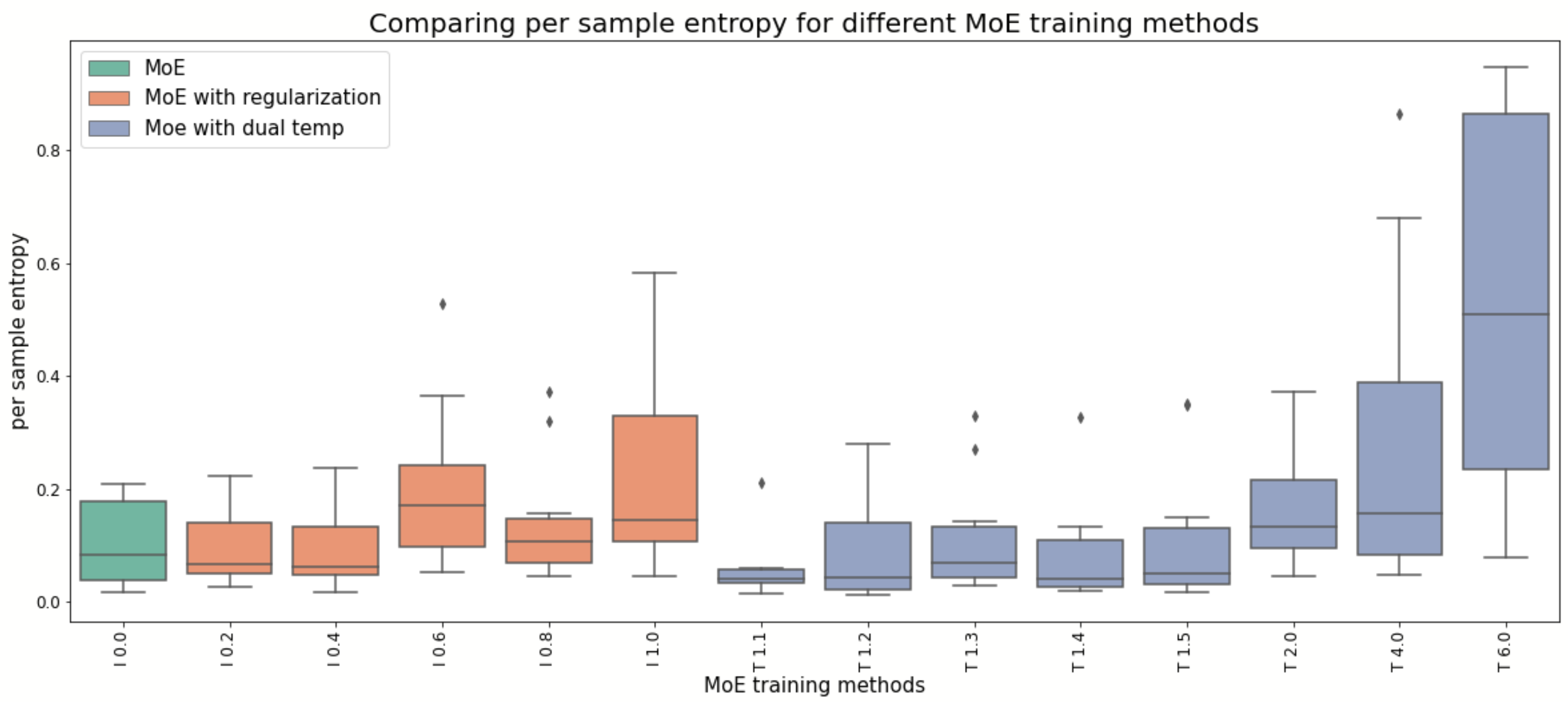
Comparing validation error for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}$ $(\mathrm{I}) \in \{0.2, \dots, 1.0\}$ and dual temperature training with temperature $T \in \{1.1, \dots, 50\}$

Our Approach: Decoupling training of gate and experts using dual temperature softmax



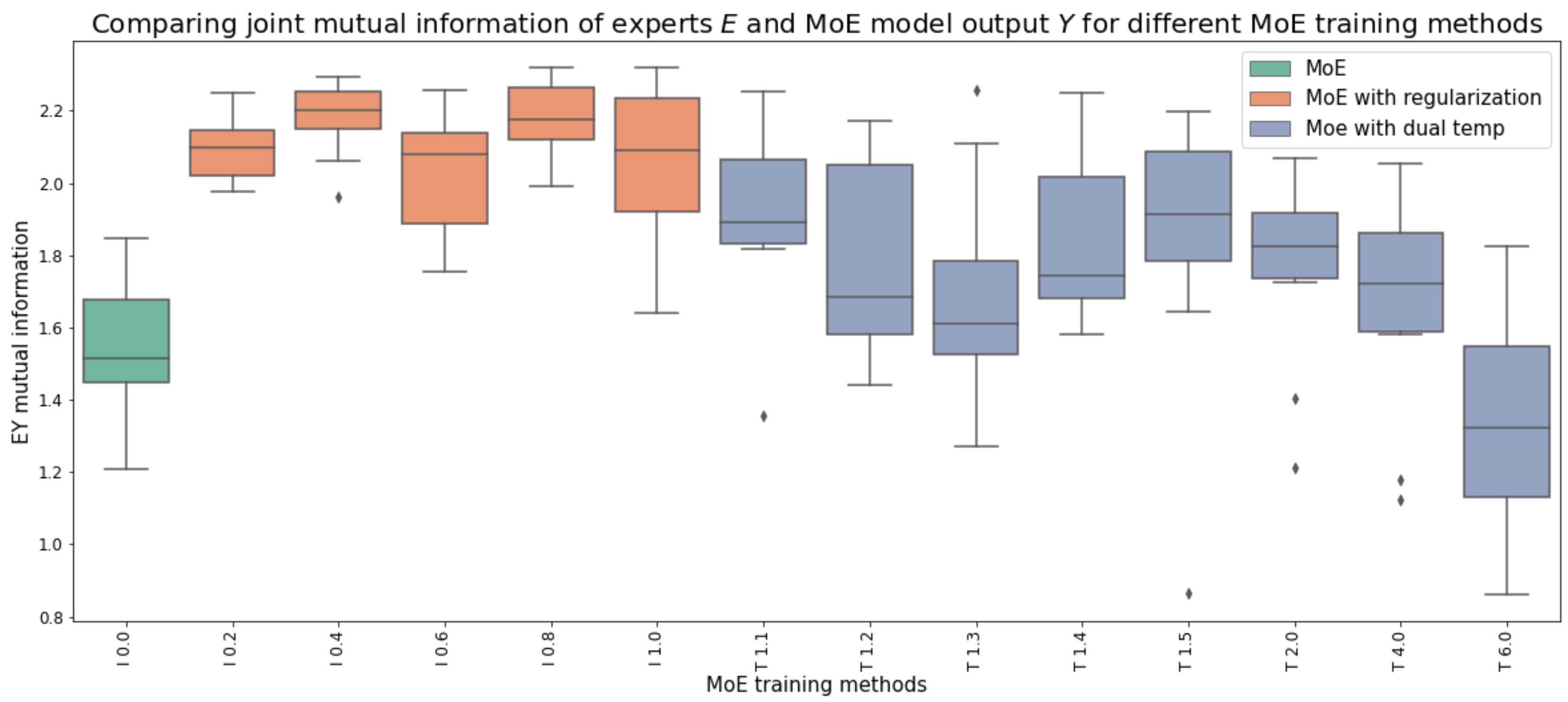
Comparing expert usage entropy for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}$ $(\mathrm{I}) \in \{0.2, \dots, 1.0\}$ and dual temperature training with temperature $T \in \{1.1, \dots, 6.0\}$

Our Approach: Decoupling training of gate and experts using dual temperature softmax



Comparing per sample expert usage entropy for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}$ $(\mathrm{I}) \in \{0.2, \ldots, 1.0\}$ and dual temperature training with temperature $T \in \{1.1, \ldots, 6.0\}$

Our Approach: Decoupling training of gate and experts using dual temperature softmax



Comparing expert and model output mutual information for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}$ (I) $\in \{0.2, \ldots, 1.0\}$ and dual temperature training with temperature $T \in \{1.1, \ldots, 6.0\}$

Our Approach: Decoupling training of gate and experts using dual temperature softmax

ROYAL HOLLOWAY UNIVERSITY OF LONDON

Observations

- Experiment results show that dual temperature training does improve expert utilization and prevents module collapse
- However, it still does not ensure an equitable expert usage
- Dual temperature training does not have a significant performance improvement over MoE with $L_{importance}$ regularization
- Hence, just an equitable distribution of samples to experts does not ensure equitable expert usage

Our Approach: Attentive Gate

Motivation

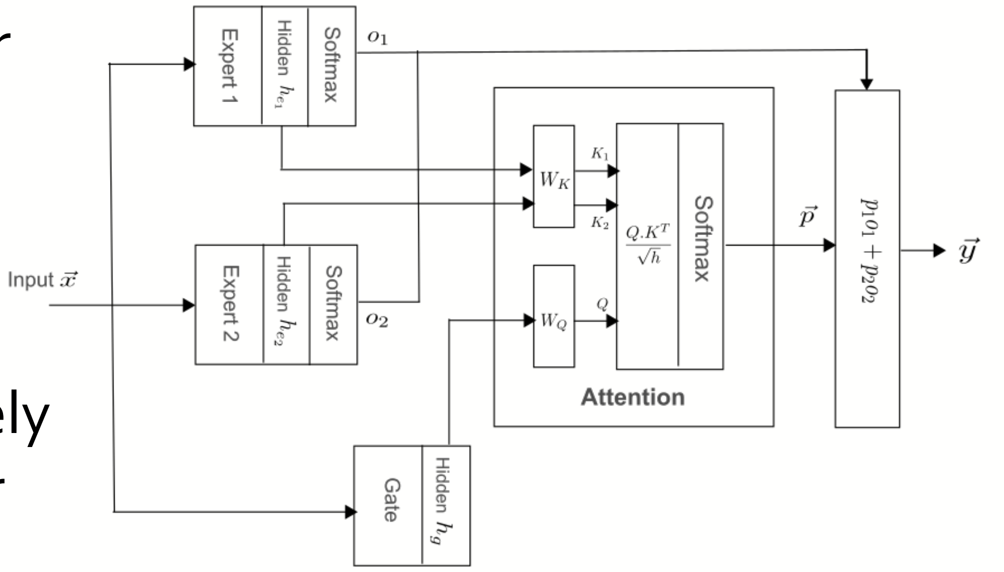
- We saw that just improving sample distribution to experts does not ensure a good gate task decomposition
- Gate's probability prediction for a given sample is based on what the gate has learnt from the experts' previous predictions for the sample but not on their current predictions for the sample
- Can we inform the gate of the current and previous predictions of the expert for the sample to make a more informed prediction?
- The attention mechanism seems like a natural choice to achieve this. The gate can learn which expert to attend to for a given sample based on its current and previous predictions for that sample

Our Approach: Attentive Gate

Method

- Query Q is computed with output of the gate hidden layer
- Keys K are computed with the outputs of the expert hidden layers
- W_Q and W_K are the query and key weight matrices respectively
- h is the size of the hidden layer
- Attention scores $A(Q, K)$ computed with expert and gate outputs are used as the gating probabilities $\vec{p}$

Attentive gate architecture with 2 experts

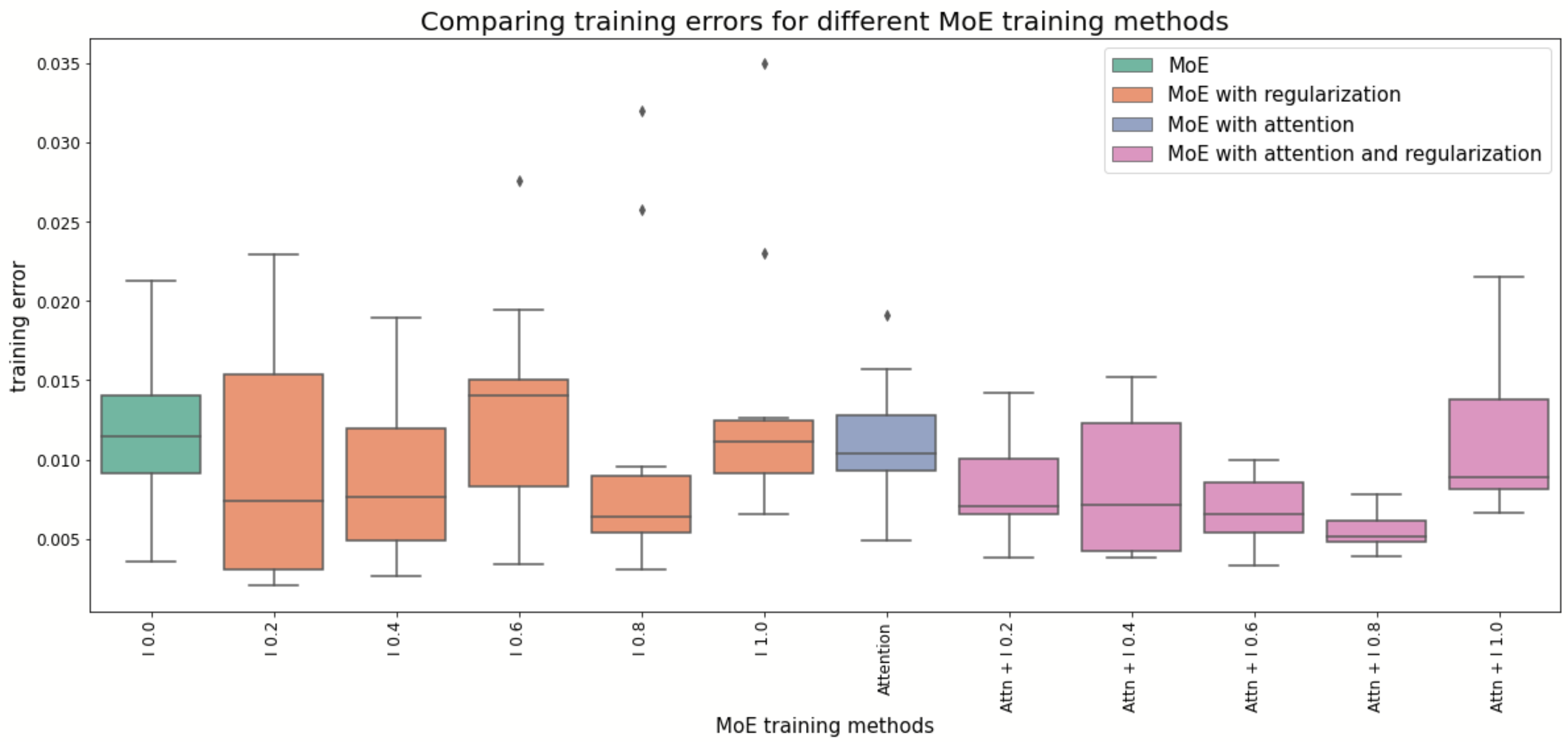


$$Q = W_Q \cdot h_g$$

$$K_i = W_K \cdot h_{e_i}$$

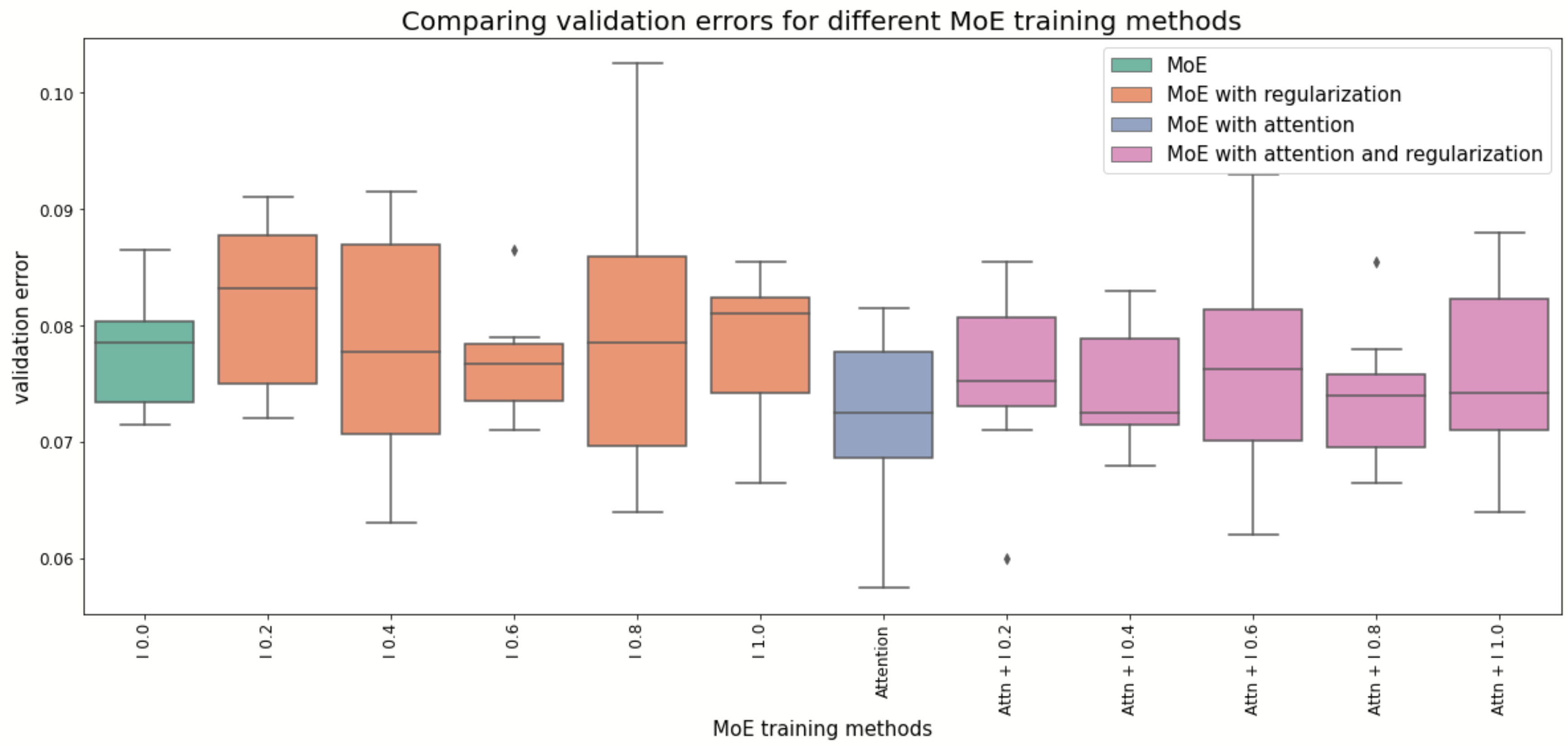
$$\vec{p} = A(Q, K) = softmax(Q \cdot K^T / \sqrt{h})$$

Our Approach: Attentive Gate



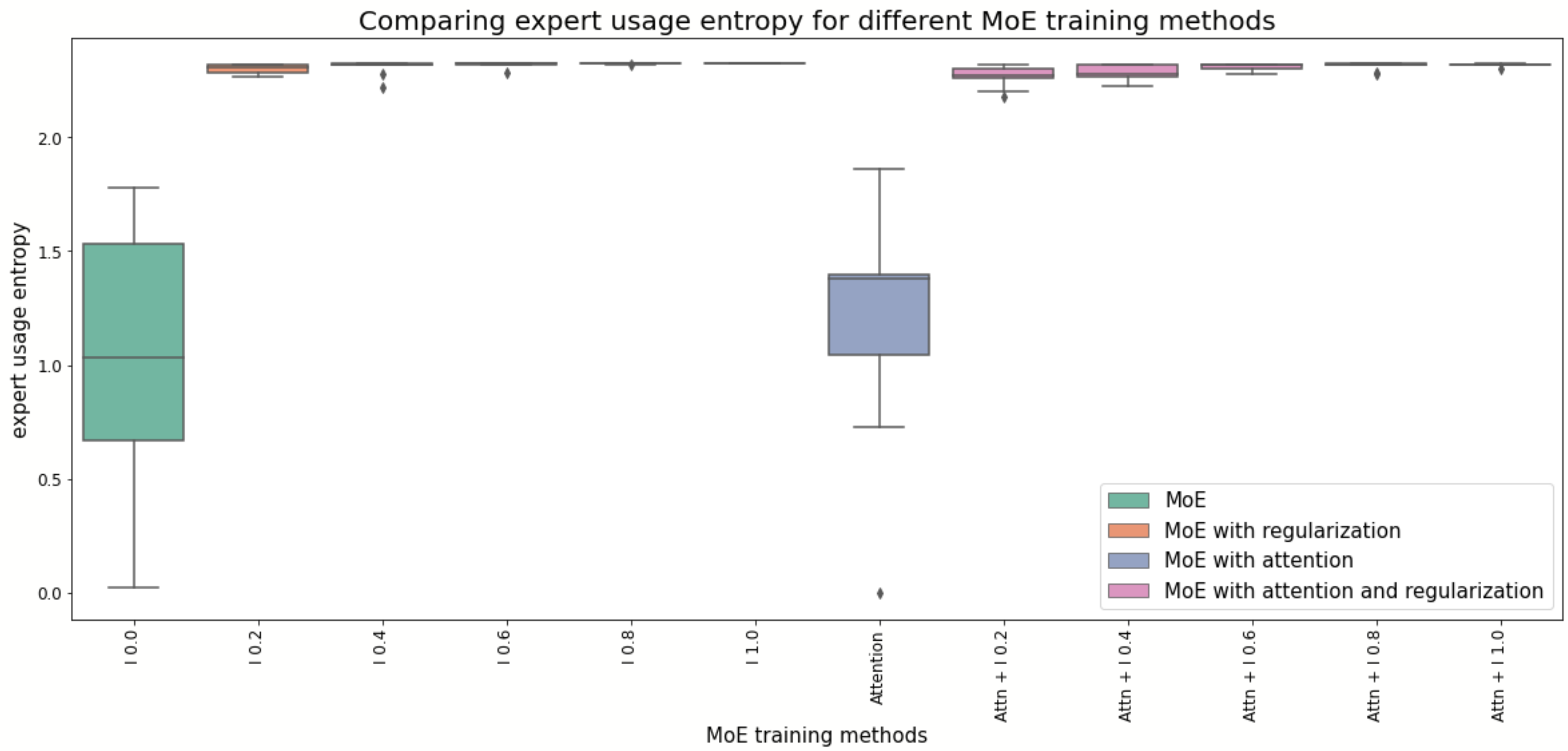
Comparing training error for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}(I) \in \{0.2, \dots, 1.0\}$, MoE training with attention, and combined training with attention and $L_{importance}$ regularization for $w_{importance}(I) \in \{0.2, \dots, 1.0\}$

Our Approach: Attentive Gate



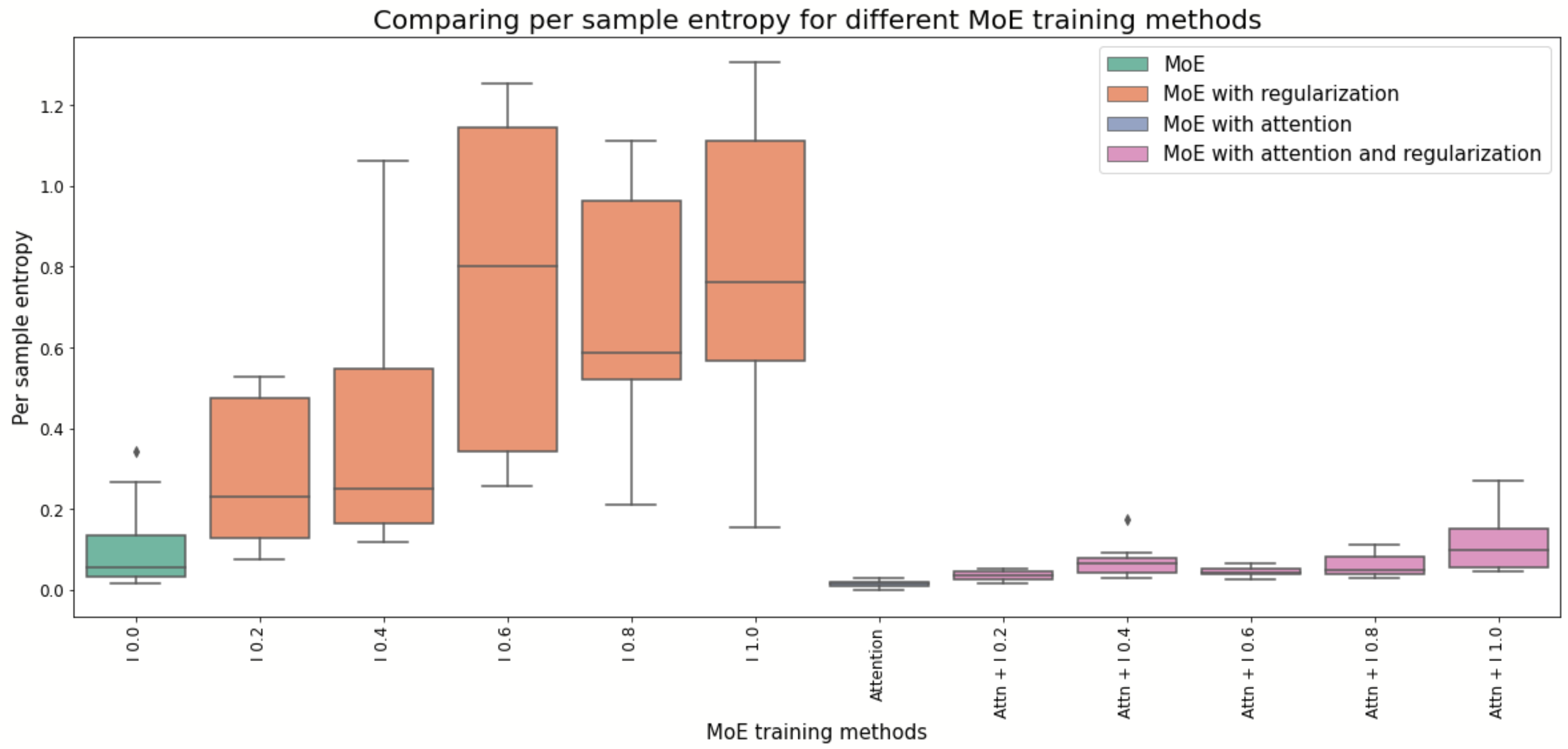
Comparing validation error for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}(I) \in \{0.2, \dots, 1.0\}$, MoE training with attention and combined training with attention and $L_{importance}$ regularization for $w_{importance}(I) \in \{0.2, \dots, 1.0\}$

Our Approach: Attentive Gate



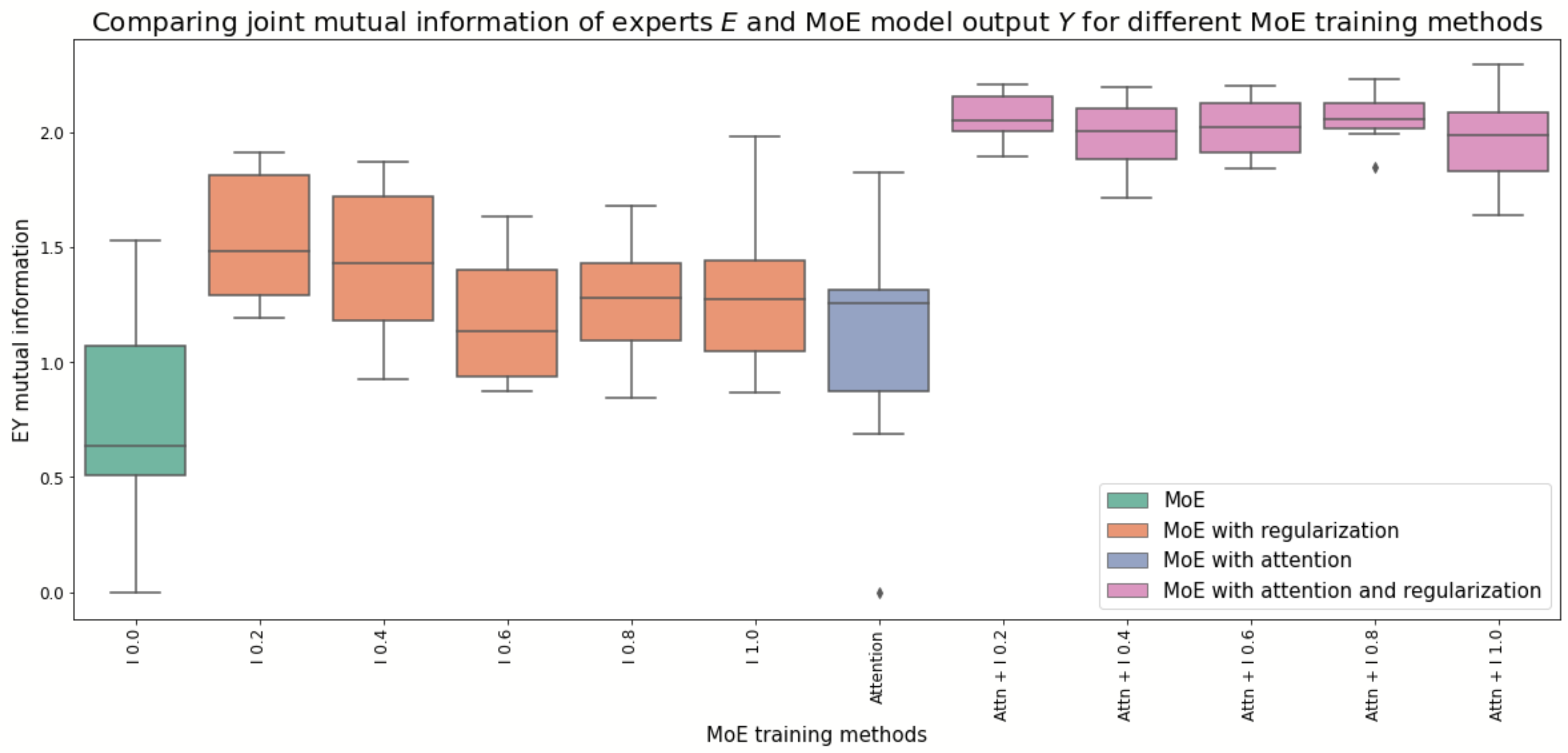
Comparing expert usage for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}(I) \in \{0.2, \ldots, 1.0\}$, MoE training with attention and combined training with attention and $L_{importance}$ regularization for $w_{importance}(I) \in \{0.2, \ldots, 1.0\}$

Our Approach: Attentive Gate



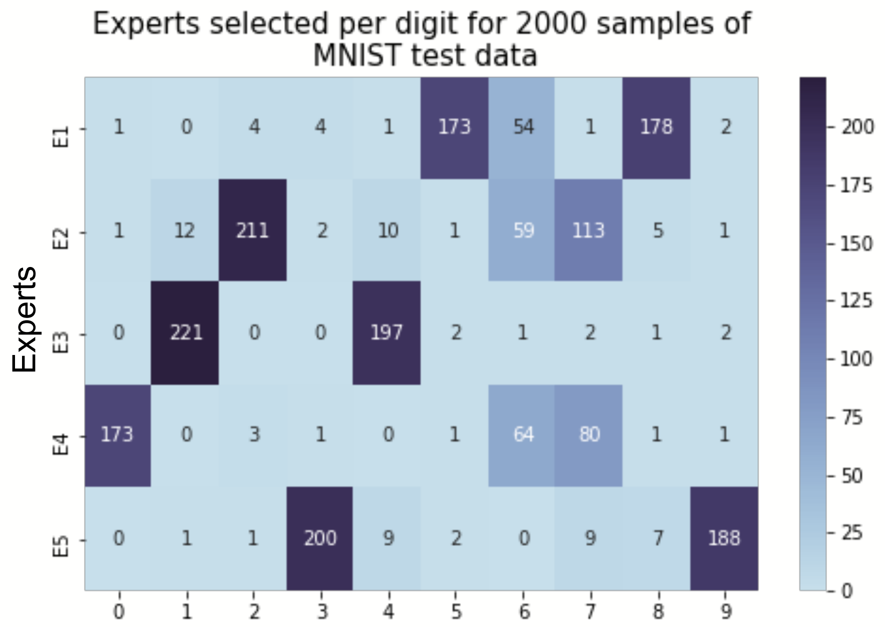
Comparing per sample expert usage entropy for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}(I) \in \{0.2, \dots, 1.0\}$, MoE training with attention and combined training with attention and $L_{importance}$ regularization for $w_{importance}(I) \in \{0.2, \dots, 1.0\}$

Our Approach: Attentive Gate

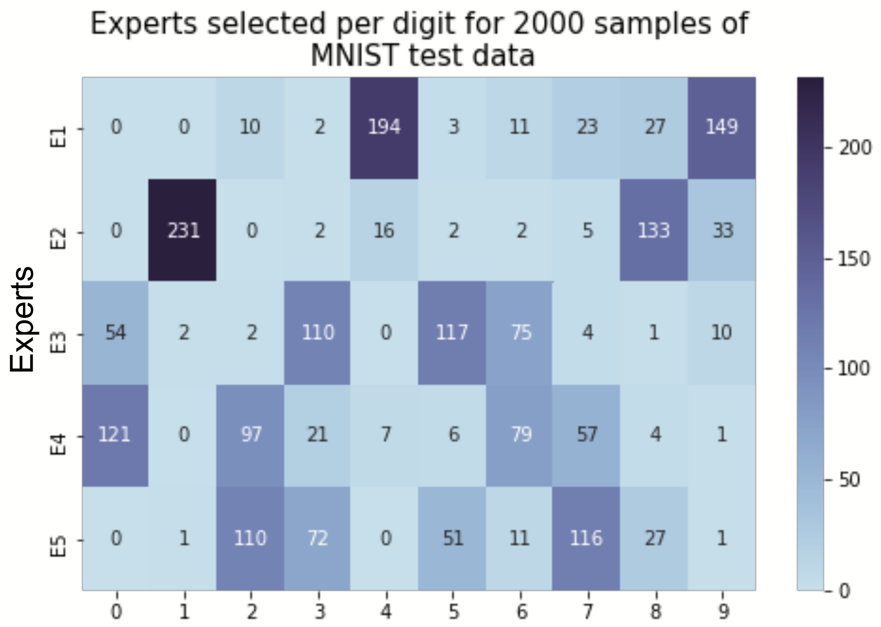


Comparing mutual information for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance}(I) \in \{0.2, \dots, 1.0\}$, MoE training with attention and combined training with attention and $L_{importance}$ regularization for for $w_{importance}(I) \in \{0.2, \dots, 1.0\}$

Our Approach: Attentive Gate



(a)



(b)

Inference on 2000 samples of MNIST test data: (a) Experts used by the attentive gate for classification of each digit with $L_{importance}$ regularization; (b) Experts used by the gate for classification of each digit with $L_{importance}$ regularization.

Our Approach: Attentive Gate

Observations

- Just the attentive gate does not ensure an equitable expert usage as it still has the gate's expert favoring problem and the initial learning of the experts
- But attentive gating has better validation error than MoE with $L_{importance}$ regularization
- However, when we also include $L_{importance}$ regularization then the combined loss training improves expert usage, per sample entropy and mutual information between expert and model output
- The task decomposition with regularized attentive gate is cleaner than with just regularization
- Currently the attention mechanism is used for both training and inference. This does not allow conditional computation during inference. So, instead we can use the trained experts to train a gate with the usual end-to-end method to achieve conditional computation during inference.

Our Approach: Sample-similarity regularization

Motivation

- We saw that just attending to the expert output does not improve the expert usage as it still depends on the initial learning of the experts
- Dual temperature training avoids expert starvation but does not guarantee good expert usage.
- The above methods and the original MoE architecture's performance are dependent on the initial learning of the expert and the gate and are prone to the initial mistakes leading to local optimums rather than global optimums
- What if we could include some domain knowledge to guide the gate and the experts during initial training? A simple heuristic for domain knowledge could be batchwise sample similarity

Our Approach: Sample-similarity regularization

ROYAL HOLLOWAY UNIVERSITY OF LONDON

Method

- We regularize the gate probabilities by preferring the same expert for similar samples and different experts for dissimilar samples
- The assumption here is that similar samples belong to similar tasks
- We define the regularization loss $L_{similarity}$ as:

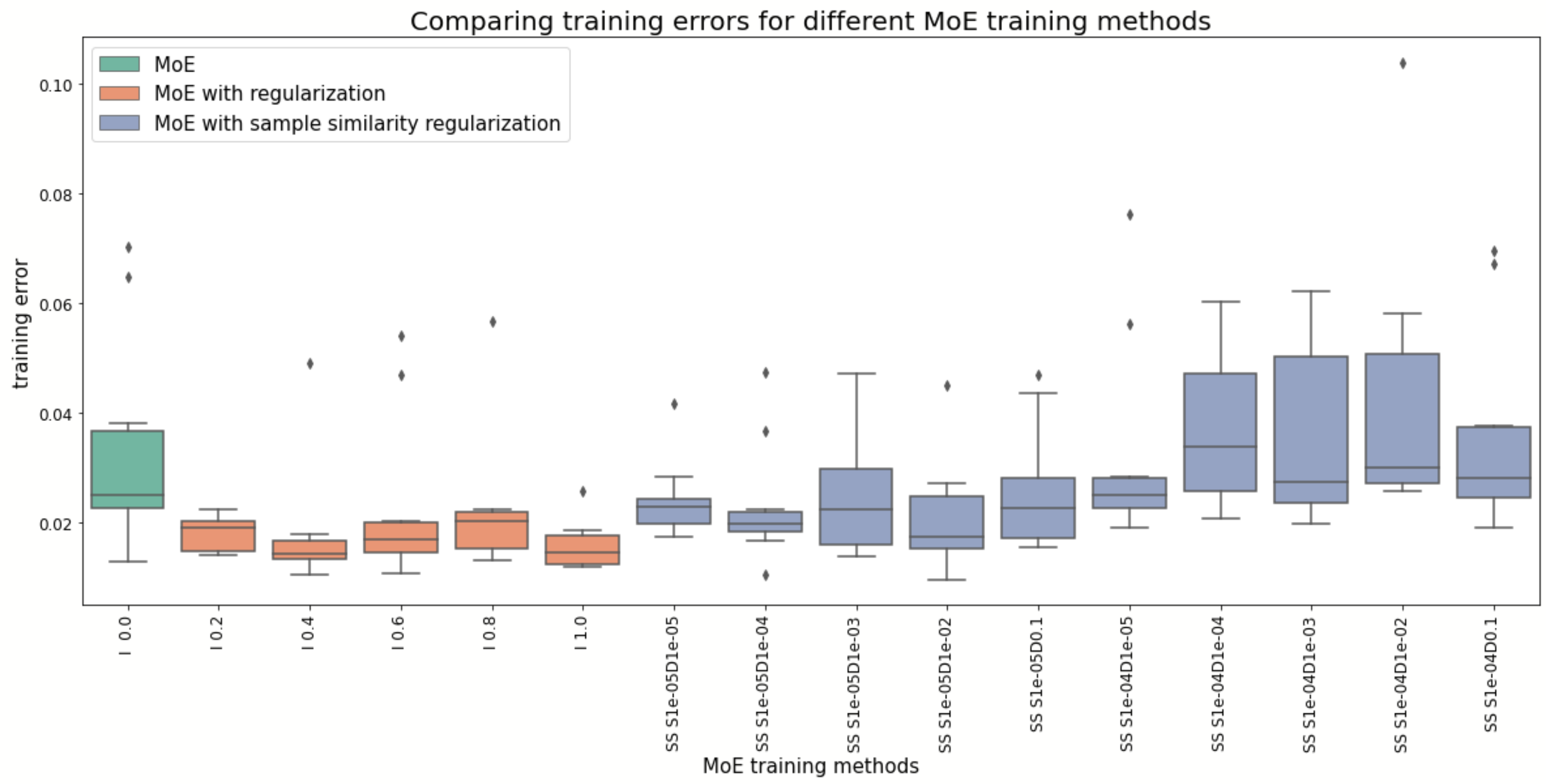
$$L_{similarity} = \frac{1}{(B^2-B)}\left[\sum_{x,x'} \frac{1}{K}\sum_e \lambda_{same}\, p(e|x)p(e|x')||x-x'||_2 - \frac{1}{(K^2-K)}\sum_{e,e'} \lambda_{diff}\, p(e|x)p(e'|x')||x-x'||_2\right]$$

where B is batch size; K is the number of experts; $x, x' \in X$ are pairs of samples in batch X; $e, e' \in E_K$ are the experts assigned to x, x' respectively and $\lambda_{same}, \lambda_{diff}$ are hand crafted parameters

- We use Euclidean distance in the above formulation but we can use other measures such as an RBF kernel.

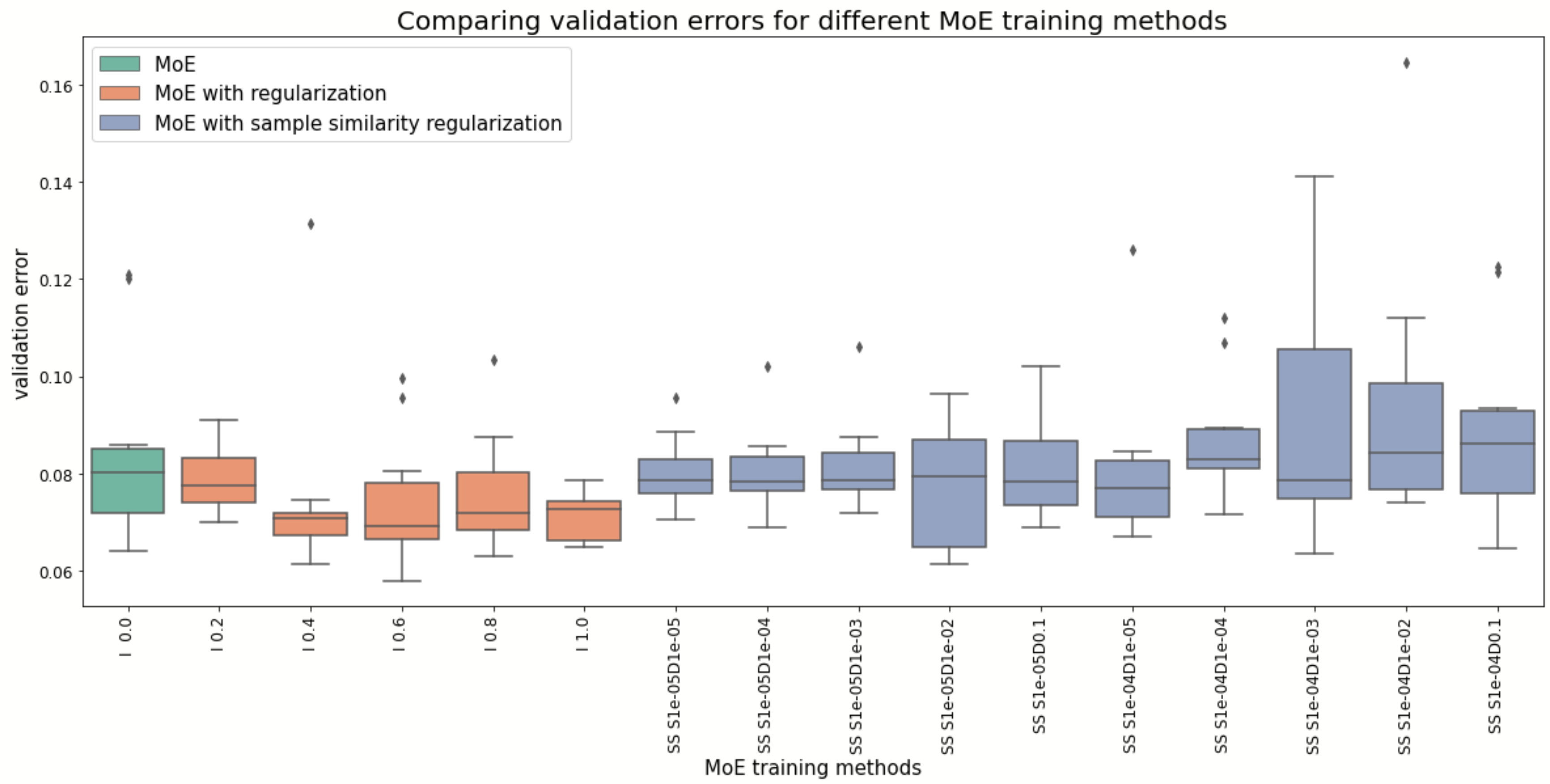
$$\text{Loss} = -\sum_{i=1}^{M} T_i \log P_i \; (cross\ entropy\ loss) + \frac{1}{(B^2-B)}\left[\sum_{x,x'} \frac{1}{K}\sum_e \lambda_{same} p(e|x)p(e|x')||x-x'||_2 - \frac{1}{(K^2-K)}\sum_{e,e'} \lambda_{diff} p(e|x)p(e'|x')||x-x'||_2\right] \; (regularization)$$

Our Approach: Sample-similarity regularization



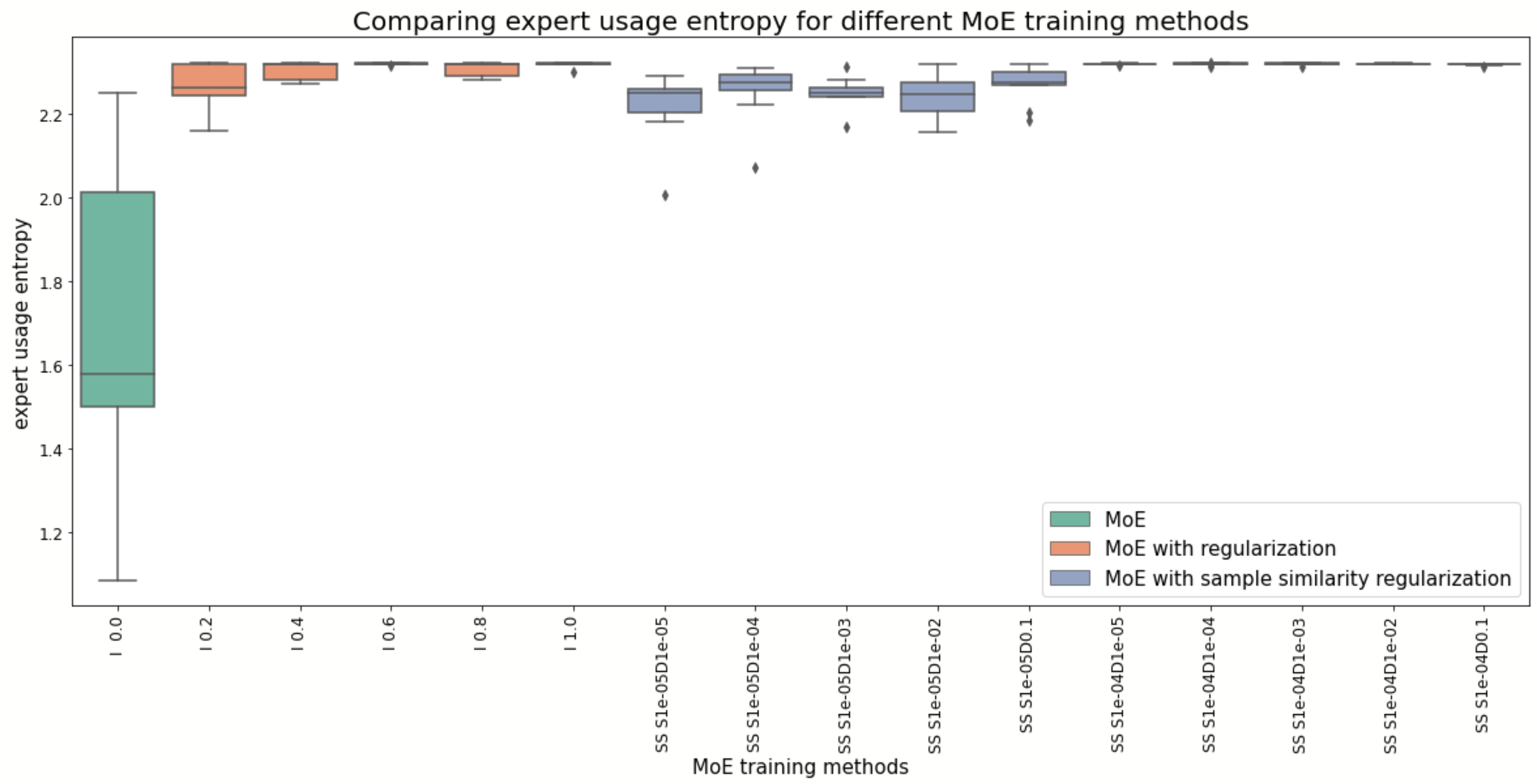
Comparing training error for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance} \in \{0.2, \ldots, 1.0\}$, MoE with $L_{similarity}$ regularization regularization for $\lambda_{same} \in \{10^{-5}, 10^{-4}\}, \lambda_{diff} \in \{10^{-5}, \ldots, 10^{-1}\}$

Our Approach: Sample-similarity regularization



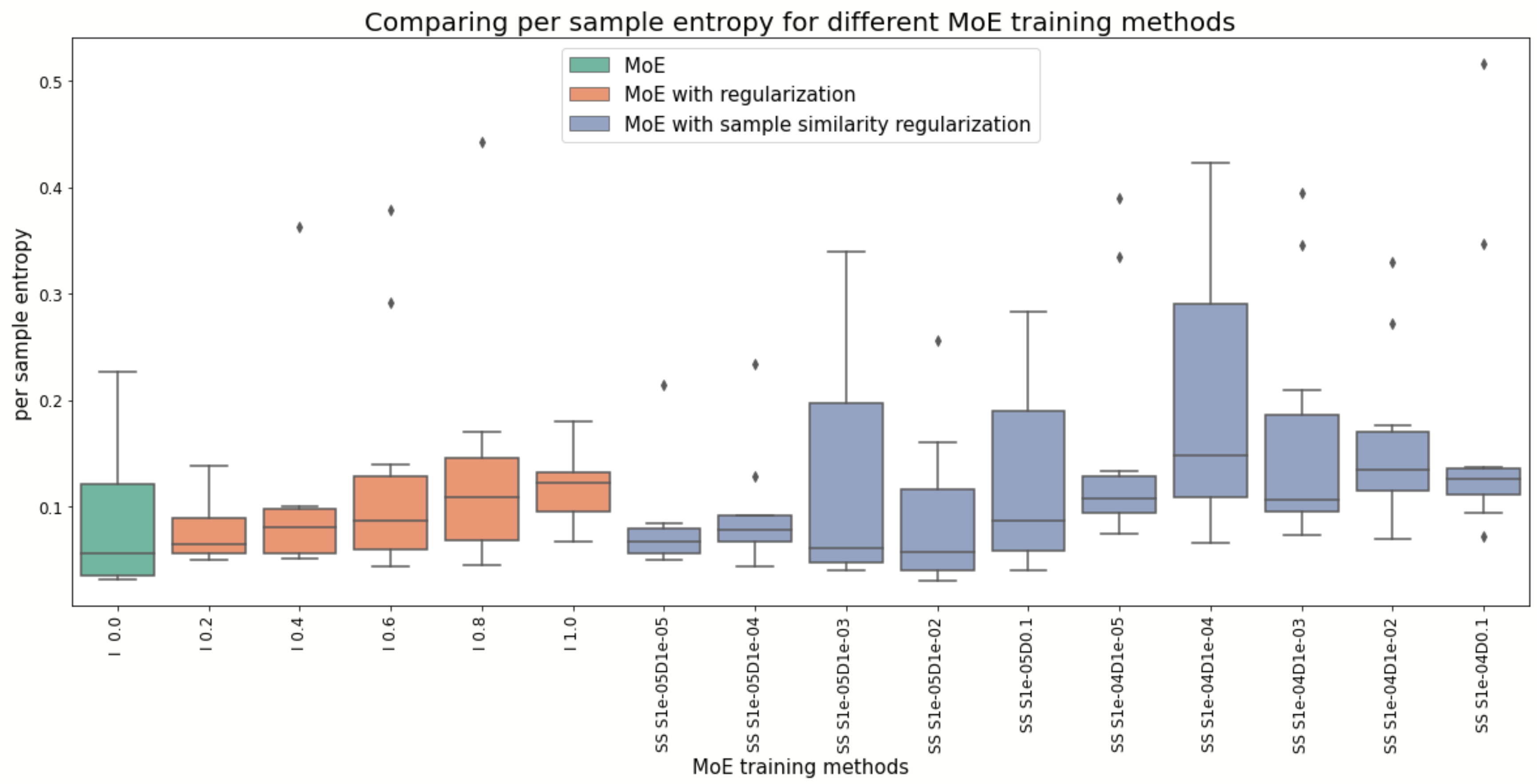
Comparing validation error for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance} \in \{0.2, \dots, 1.0\}$, MoE with $L_{similarity}$ regularization regularization for $\lambda_{same} \in \{10^{-5}, 10^{-4}\}, \lambda_{diff} \in \{10^{-5}, \dots, 10^{-1}\}$

Our Approach: Sample-similarity regularization



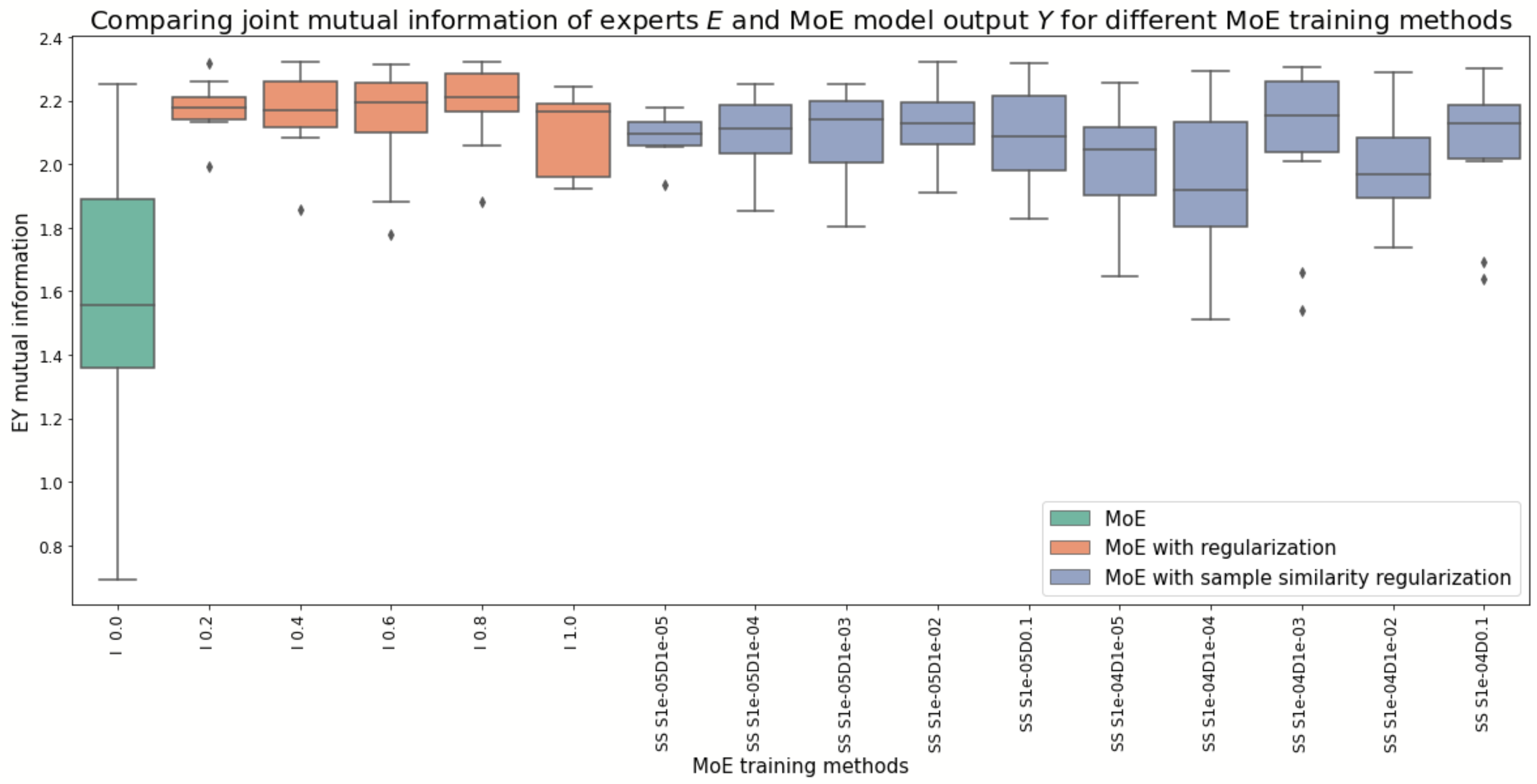
Comparing expert usage for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance} \in \{0.2, \dots, 1.0\}$, MoE with $L_{similarity}$ regularization regularization for $\lambda_{same} \in \{10^{-5}, 10^{-4}\}, \lambda_{diff} \in \{10^{-5}, \dots, 10^{-1}\}$

Our Approach: Sample-similarity regularization



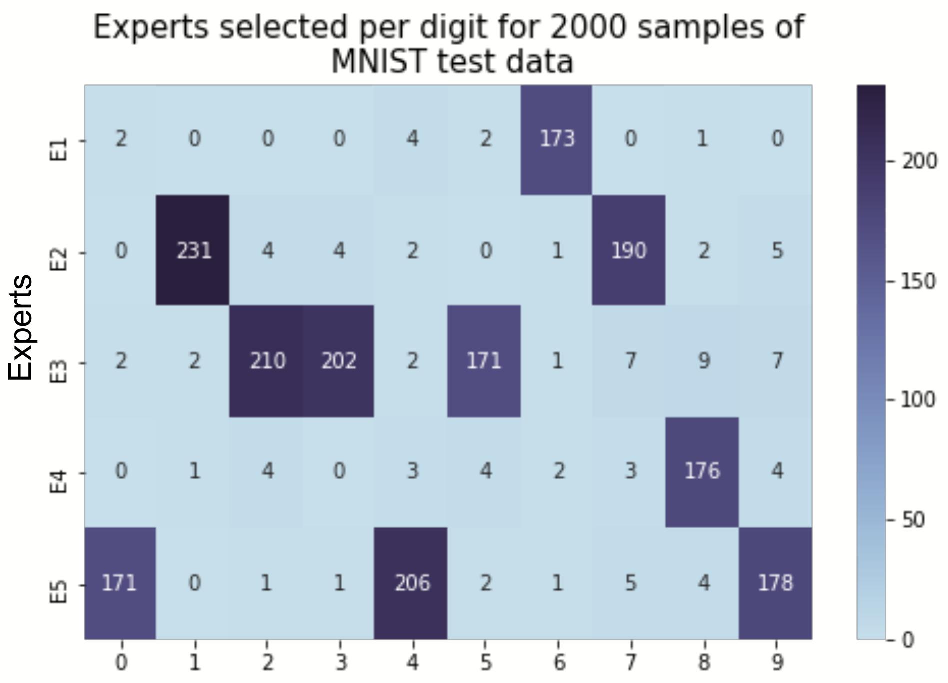
Comparing sample entropy for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance} \in \{0.2, \ldots, 1.0\}$, MoE with $L_{similarity}$ regularization regularization for $\lambda_{same} \in \{10^{-5}, 10^{-4}\}, \lambda_{diff} \in \{10^{-5}, \ldots, 10^{-1}\}$

Our Approach: Sample-similarity regularization

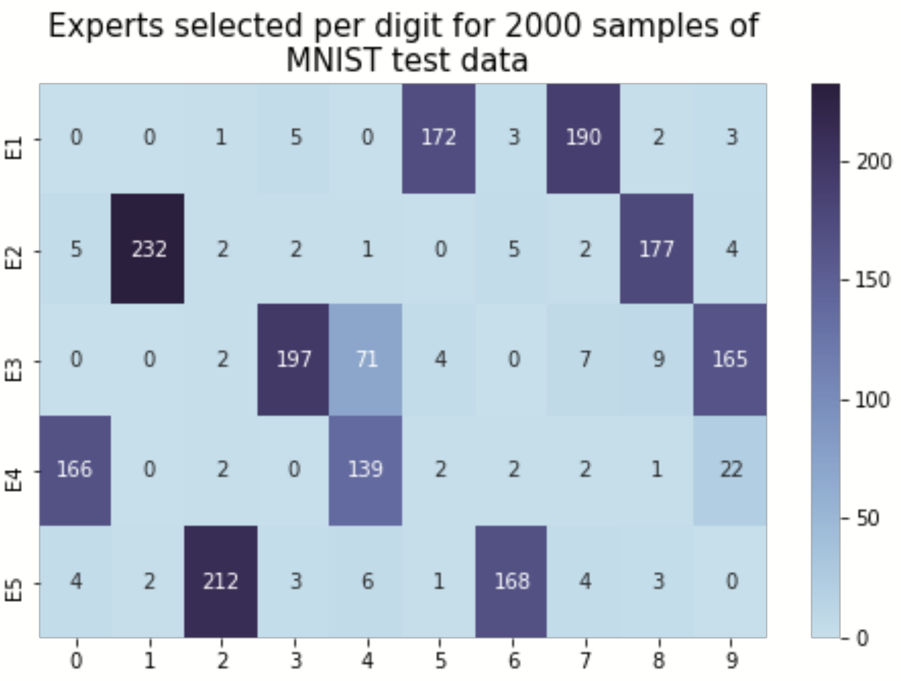


Comparing mutual information for MoE without regularization, MoE with $L_{importance}$ regularization for $w_{importance} \in \{0.2, \dots, 1.0\}$, MoE with $L_{similarity}$ regularization regularization for $\lambda_{same} \in \{10^{-5}, 10^{-4}\}, \lambda_{diff} \in \{10^{-5}, \dots, 10^{-1}\}$

Our Approach: Sample-similarity regularization



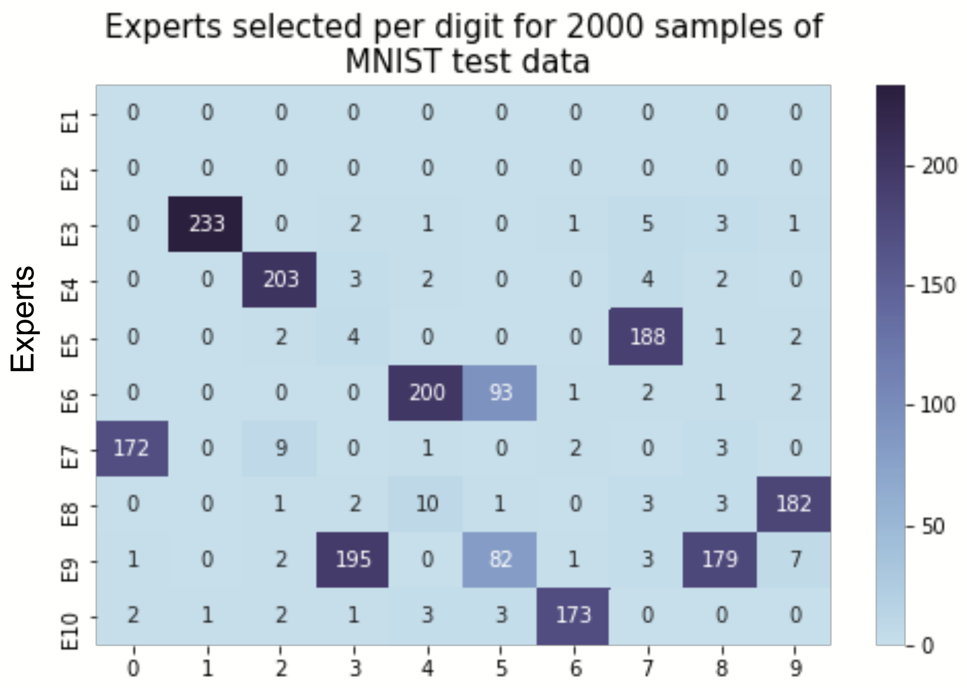
(a)



(b)

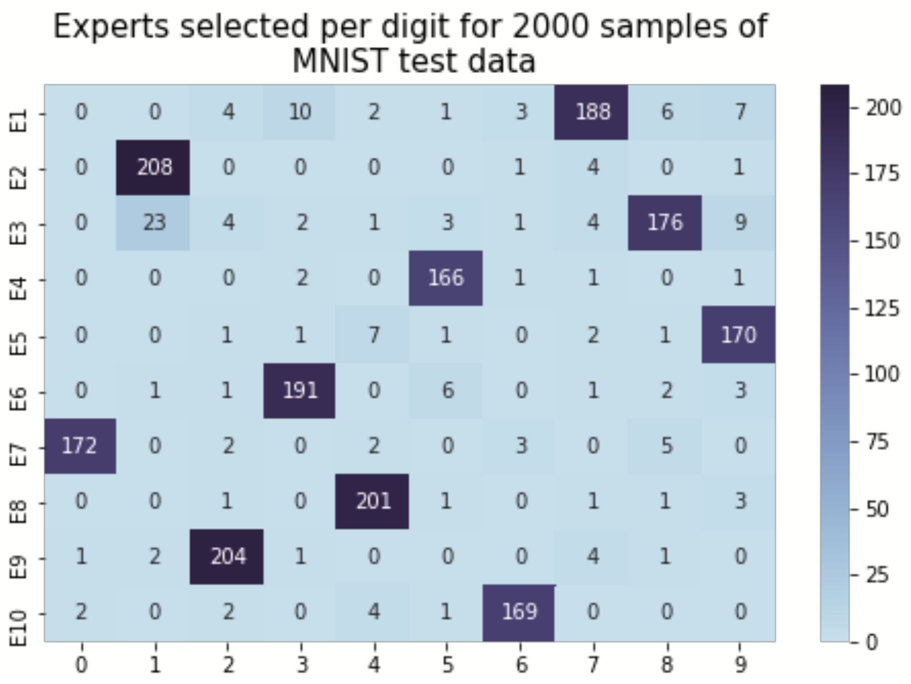
Inference on 2000 samples of MNIST test data: (a) Experts used by gate for classification of each digit with $L_{similarity}$ regularization; (b) Experts used by the gate for classification of each digit with $L_{importance}$ regularization.

Our Approach: Sample-similarity regularization



Experts	0	1	2	3	4	5	6	7	8	9
E1	0	0	0	0	0	0	0	0	0	0
E2	0	0	0	0	0	0	0	0	0	0
E3	0	233	0	2	1	0	1	5	3	1
E4	0	0	203	3	2	0	0	4	2	0
E5	0	0	2	4	0	0	0	188	1	2
E6	0	0	0	0	200	93	1	2	1	2
E7	172	0	9	0	1	0	2	0	3	0
E8	0	0	1	2	10	1	0	3	3	182
E9	1	0	2	195	0	82	1	3	179	7
E10	2	1	2	1	3	3	173	0	0	0

(a)

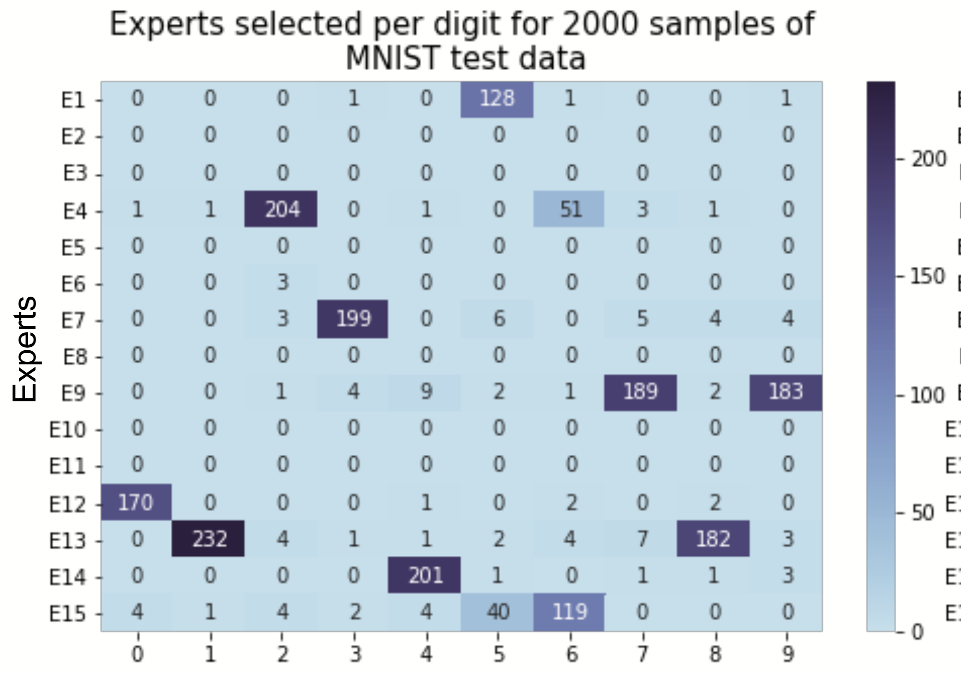


	0	1	2	3	4	5	6	7	8	9
E1	0	0	4	10	2	1	3	188	6	7
E2	0	208	0	0	0	0	1	4	0	1
E3	0	23	4	2	1	3	1	4	176	9
E4	0	0	0	2	0	166	1	1	0	1
E5	0	0	1	1	7	1	0	2	1	170
E6	0	1	1	191	0	6	0	1	2	3
E7	172	0	2	0	2	0	3	0	5	0
E8	0	0	1	0	201	1	0	1	1	3
E9	1	2	204	1	0	0	0	4	1	0
E10	2	0	2	0	4	1	169	0	0	0

(b)

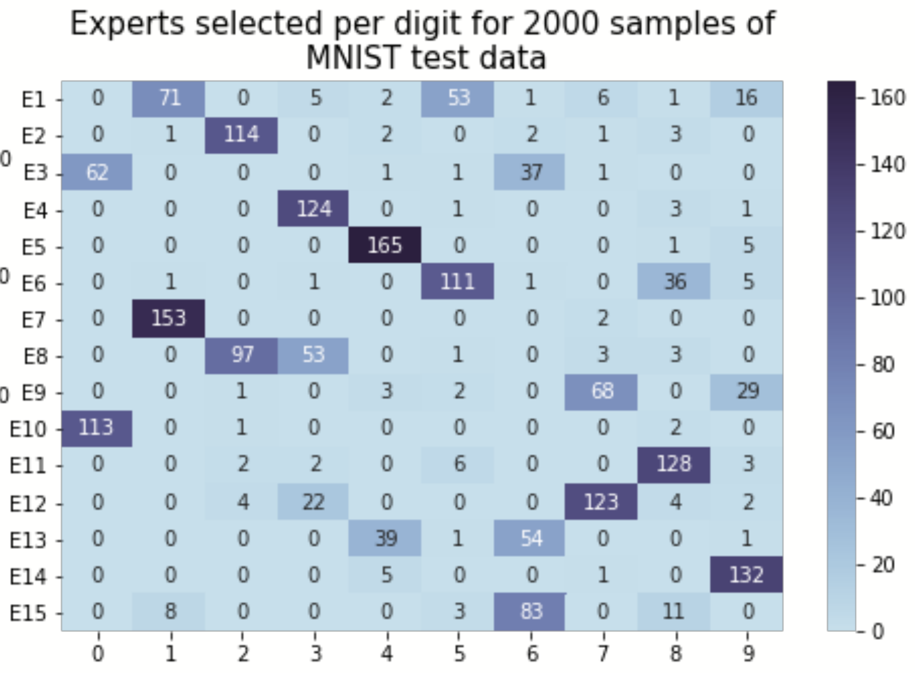
Inference on 2000 samples of MNIST test data with 10 experts : (a) Experts used by gate for classification of each digit with $L_{similarity}$ regularization; (b) Experts used by the gate for classification of each digit with $L_{importance}$ regularization.

Our Approach: Sample-similarity regularization



	0	1	2	3	4	5	6	7	8	9
E1	0	0	0	1	0	128	1	0	0	1
E2	0	0	0	0	0	0	0	0	0	0
E3	0	0	0	0	0	0	0	0	0	0
E4	1	1	204	0	1	0	51	3	1	0
E5	0	0	0	0	0	0	0	0	0	0
E6	0	0	3	0	0	0	0	0	0	0
E7	0	0	3	199	0	6	0	5	4	4
E8	0	0	0	0	0	0	0	0	0	0
E9	0	0	1	4	9	2	1	189	2	183
E10	0	0	0	0	0	0	0	0	0	0
E11	0	0	0	0	0	0	0	0	0	0
E12	170	0	0	0	1	0	2	0	2	0
E13	0	232	4	1	1	2	4	7	182	3
E14	0	0	0	0	201	1	0	1	1	3
E15	4	1	4	2	4	40	119	0	0	0

(a)



	0	1	2	3	4	5	6	7	8	9
E1	0	71	0	5	2	53	1	6	1	16
E2	0	1	114	0	2	0	2	1	3	0
E3	62	0	0	0	1	1	37	1	0	0
E4	0	0	0	124	0	1	0	0	3	1
E5	0	0	0	0	165	0	0	0	1	5
E6	0	1	0	1	0	111	1	0	36	5
E7	0	153	0	0	0	0	0	2	0	0
E8	0	0	97	53	0	1	0	3	3	0
E9	0	0	1	0	3	2	0	68	0	29
E10	113	0	1	0	0	0	0	0	2	0
E11	0	0	2	2	0	6	0	0	128	3
E12	0	0	4	22	0	0	0	123	4	2
E13	0	0	0	0	39	1	54	0	0	1
E14	0	0	0	0	5	0	0	1	0	132
E15	0	8	0	0	0	3	83	0	11	0

(b)

Inference on 2000 samples of MNIST test data with 15 experts : (a) Experts used by gate for classification of each digit with $L_{similarity}$ regularization; (b) Experts used by the gate for classification of each digit with $L_{importance}$ regularization.

Our Approach: Sample-similarity regularization

Observations

- $L_{similarity}$ regularization distributes the samples equitably between the experts with similar samples routed to the same expert
- Unlike $L_{importance}$, $L_{similarity}$ only uses experts required for the tasks and not all experts equally
- $L_{similarity}$ performs as well as $L_{importance}$ regularization for low values of λ_{same} and λ_{diff}
- It provides a flexible method to include domain knowledge for task decomposition

Summary

- We present experiments to better understand how the gate decomposes the tasks between modules in MoE.
- Our experiments revealed that:
  - MoEs are indeed inherently interpretable, however,
  - Existing architectures and methods of training them do not necessarily guarantee an interpretable gate decomposition among the modules.
  - There is no learning or error disadvantage for the gate to learn an interpretable task decomposition.
  - The heuristic of jointly optimising the gate and modules leads to uninterpretable task decompositions.
- Achieving an interpretable task decomposition between the modules
  - can allow error attribution to either the gate or the modules, and
  - Additionally the modules would be better suited for transferability
- We proposed 3 methods to improve batch distribution and expert usage

Future Work

- Test our approach on more complex and bigger datasets and architectures
- Currently the gate learns the expert selection policy. What if instead it learns to predict the expert loss?
- Address the following research questions
  - Formalise gate and expert error attribution
  - Does the same decomposition of the tasks enable both interpretability and transferability? Or do they need different task decompositions?

Thank You!

ROYAL HOLLOWAY UNIVERSITY OF LONDON

References

1. N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017.
2. Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale. ArXiv, January 2022
3. L. Kaiser, A. N. Gomez, N. Shazeer, A. Vaswani, N. Parmar, L. Jones, and J. Uszkoreit. One model to learn them all. CoRR, abs/1706.05137, 2017
4. H. Hihn and D. A. Braun. Mixture-of-variational-experts for continual learning. In ICLR Workshop on Agent Learning in Open-Endedness, 2022.
5. R. Jacobs, M. Jordan, S. Nowlan, and G. Hinton. Adaptive mixture of local expert. Neural Computation, 3:78–88, 02 1991
6. R. A. Jacobs, M. I. Jordan, and A. G. Barto. Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. *Cognitive Science*", 15(2):219 – 250, 1991.
7. M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. Proceedings of the International Joint Conference on Neural Networks, 2:1339–1344, 1993.

References

8. L. Kirsch, J. Kunze, and D. Barber. Modular networks: Learning to de-compose neural computation. In Advances in Neural Information Processing Systems, volume 31. Curran Associates, Inc.,
2018.

9. G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network, 2015. NIPS 2014 Deep Learning Workshop