Towards Understanding MoE

An MoE layer contains many experts that share the same network architecture and are trained by the same algorithm, with a gating/routing function that routes individual inputs to a few experts among all the candidates.

The number of experts used for an input can be a hyperparameter choice called top-k (usually 1 or 2). The computation (inference) cost will only be the computation cost for the top-k expert(s) used.

In practice, all experts are initialized with the same weight distribution, optimization configuration, and the router is configured to distribute the data evenly between experts (traditionally through random noise and/or an auxiliary load balancing loss). This makes it unclear how this leads to specialization of each expert, instead of collapsing into a single model.

Key findings:

- MoE with linear experts cannot be trained to find a good classifier efficiently. An MoE with non-linear experts trained with gradient descent from random initialization can accomplish this. The gating mechanism, however, can be linear, since it only needs to differentiate between input clusters.
- The study shows that adding random noise to the router’s choice in soft routing (before the discrete choice) helps distribute the data across experts.

- For nonlinear MoE with non-linear expert functions, experts will diverge at the end of the exploration stage. At the end of the exploration stage, an expert will achieve low error in a specific cluster, but high error on the other clusters.
- There is a potential load unbalancing issue when training MoE, with the probability of each input being routed to the same few experts being high. This is a self-fulfilling prophecy, as it will lead to more training of these few experts, resulting in a bigger imbalance. Normalized gradient descent can help with this issue, as well as adding a penalty term to the loss function (auxiliary load balancing loss) or random noise to the router.
- The advantage of MoE over dense models in terms of performance depends on the task and the cluster structure of the data.

My takeaway(s):

- In MoE, the router specializes in dividing the input space into n parts/clusters (where n is the number of experts). Each expert then becomes a specialist on a specific cluster of the input space (as divided by the router).
- The router's task can be performed linearly, as it only needs to learn how to divide the input space into clusters, while the expert's task is more challenging, benefitting from non-linearities.
- It is important to employ load balancing strategies to ensure that this clustering is done correctly, especially at early stages of training when the clusters are not yet clear. If this

is not done, it can lead to generalization (some experts being assigned to large areas of the input space while others are assigned to too small areas).

- The advantages of MoE will, therefore, depend on the input space of the data – if the data can be clustered into “specialization” areas, MoE will perform better, otherwise if the task benefits from a generalized knowledge of the input space, a dense model will outperform MoE.