IIT Kanpur

CS787 - Generative AI

Final Project Report

Enhancing Switch Transformer with Adaptive Capacity, Reassignment Routing, and Expert Diversity Regularization

Student Name: Ajay Singh (220090), Lingala Adithya (220587)

Instructor: Prof. Arnab Bhattacharya, Prof .Subhajit Roy

Contents

1 Introduction

This report presents our enhancements to the Mixture-of-Experts (MoE) architecture used in the Switch Transformer. The baseline Switch Transformer routes each token to a single expert (Top-1 routing), which enables efficient large-scale sparse models. However, it still suffers from issues such as token dropping, unbalanced expert usage, and expert collapse.

To address these limitations, we propose three key innovations:

- Token-Importance Aware Routing (TIA-R)
- Adaptive Capacity with No-Token-Left-Behind (NTLB) Reassignment
- SimBal (Similarity Balancing) Router Regularization

All innovations are implemented in `switch_moe_innovations.py`.

2 Innovation 1: Token-Importance Aware Routing (TIA-R)

2.1 Motivation

The baseline router treats all tokens equally, even though some tokens (rare words, boundaries, high-activation tokens) are more important. Dropping such tokens harms training, gradient flow, and perplexity.

2.2 Mathematical Formulation

For a token representation $x \in \mathbb{R}^d$, router logits are:

$$l = W_r x$$

We compute token importance using the L2 norm:

$$s = \|x\|_2$$

Normalize importance:

$$\tilde{s} = \frac{s - \mu_s}{\sigma_s + 10^{-6}}$$

Modified logits:

$$l' = l + \lambda \cdot \tilde{s}$$

Where:

- λ: importance scaling factor
- Important tokens get higher logits $\rightarrow$ less chance of being dropped.

2.3 Code Snippet

```
imp = x.norm(p=2, dim=-1)
imp_norm = (imp - imp.mean()) / (imp.std() + 1e-6)
logits = logits + importance_lambda * imp_norm.unsqueeze(-1)
```

3 Innovation 2: Adaptive Capacity + No-Token-Left-Behind Reassignment

3.1 Problem in Baseline

Switch Transformer drops tokens when an expert exceeds its capacity:

$$C = \phi \cdot \frac{T}{E}$$

where ϕ is the capacity factor.

This leads to:

- Loss of training signal
- Higher perplexity
- Unused expert capacity in other experts

3.2 Adaptive Capacity

Let:

$$n_i = \text{tokens routed to expert } i$$
$$\mu_n = \frac{T}{E}$$

We define:

$$\Delta_i = \max\left(0, k(n_i - \mu_n)\right)$$

Thus final capacity:

$$C_i = C_0 + \Delta_i$$

This gives overloaded experts extra room while preventing unnecessary dropping.

3.3 NTLB Reassignment

For overflow tokens, we compute a reassignment score:

$$\text{score}(t, j) = \frac{p(t, j)}{1 + \text{load}(j)}$$

Token is reassigned to an expert with:

- Spare capacity
- Maximum score

If no expert is available → token is dropped (rare in practice).

3.4 Code Snippet

```
candidate_scores = (probs / (1.0 + counts_float)).detach().cpu().numpy()
best_j = argmax(candidate_scores where spare[j] > 0)
```

4 Innovation 3: SimBal (Similarity Balancing) Regularization

4.1 Motivation

Router weight matrix contains a vector per expert. If experts become similar, they collapse and specialization is lost.

4.2 Mathematical Form

Let:

$$G = WW^T$$

SimBal penalizes off-diagonal terms:

$$\mathcal{L}_{\text{simbal}} = \gamma \sum_{i \neq j} G_{ij}^2$$

This pushes experts to become diverse and orthogonal.

4.3 Code Snippet

```
W = self.switch.weight
G = W @ W.t()
off_diag = G - torch.diag(torch.diag(G))
simbal_loss = simbal_coef * (off_diag**2).sum()
```

5 Experimental Results

We trained both the baseline and the innovation model on Tiny Shakespeare for 4 epochs.

5.1 Baseline Results

(Extracted from the midsem PDF)

Epoch	Train Loss	Val Loss	Drop Rate
1	2.05	1.976	0.0308
2	1.99	1.958	0.0233
3	1.98	1.951	0.0201
4	1.97	1.942	0.0181

5.2 Innovation Model Results

(Extracted from logs)

Epoch	Train Loss	Val Loss	Drop Rate
1	2.0108	1.8781	0.0000
2	1.9024	1.8486	0.0000
3	1.8763	1.8287	0.0000
4	1.8710	1.8129	0.0000

5.3 Result Analysis

- Validation loss improved from **1.942** (baseline) to **1.8129** (innovation).
- Training loss also improved consistently.
- Drop rate reduced from **1.8%** to **0%**.
- Adaptive capacity and reassignment eliminate unnecessary dropping.
- SimBal encourages better expert specialization.

6 Conclusion

Our innovations significantly improved Switch Transformer training stability and performance.

- **Token Importance Routing** improves handling of semantically strong tokens.
- **Adaptive Capacity + Reassignment** eliminates token drop rate.
- **SimBal Regularization** enhances expert diversity and reduces collapse.

Future work includes testing on larger datasets (C4) and scaling to deeper architectures.