Report: A Hybrid ELF-ResLT Model for Long-Tailed Recognition

Objective

The primary goal was to create a novel hybrid neural network architecture that combines the core ideas from two state-of-the-art papers, **ELF** and **ResLT**. The aim was to develop a model that could outperform both individual methods on the challenging long-tailed CIFAR-10 dataset.

Foundational Concepts

- **ELF's Contribution:** This paper introduced the concept of **example-level hardness**. Its early-exiting framework allows the model to filter out "easy" images and focus its computational resources on the "hard" ones.
- **ResLT's Contribution:** This paper introduced **class-level re-balancing in the parameter space**. Its specialized three-branch head is explicitly designed to improve performance on the under-represented medium and tail classes.

Hyperparameter values:

Class Splits: 'many': [0, 1, 2], 'medium': [3, 4, 5], 'few': [6, 7, 8, 9]

NUM_EPOCHS = 200

BATCH_SIZE = 128

BASE_LR = 0.1

MOMENTUM = 0.9

ALPHA = 0.95

WARMUP_EPOCHS = 5

IMBALANCE_RATIO = 100/50/10

Except for Models 2,3 the training and inference threshold are constant and are as follows for the following loss function

For LDAM Loss:

TRAINING_EXIT_THRESHOLD = 0.5

INFERENCE_EXIT_THRESHOLD = 0.5

For Cross Entropy Loss:

TRAINING_EXIT_THRESHOLD = 0.9

INFERENCE_EXIT_THRESHOLD = 0.5

[Models 2:

TRAINING_EXIT_THRESHOLD = 0.2

INFERENCE_EXIT_THRESHOLD = 0.25

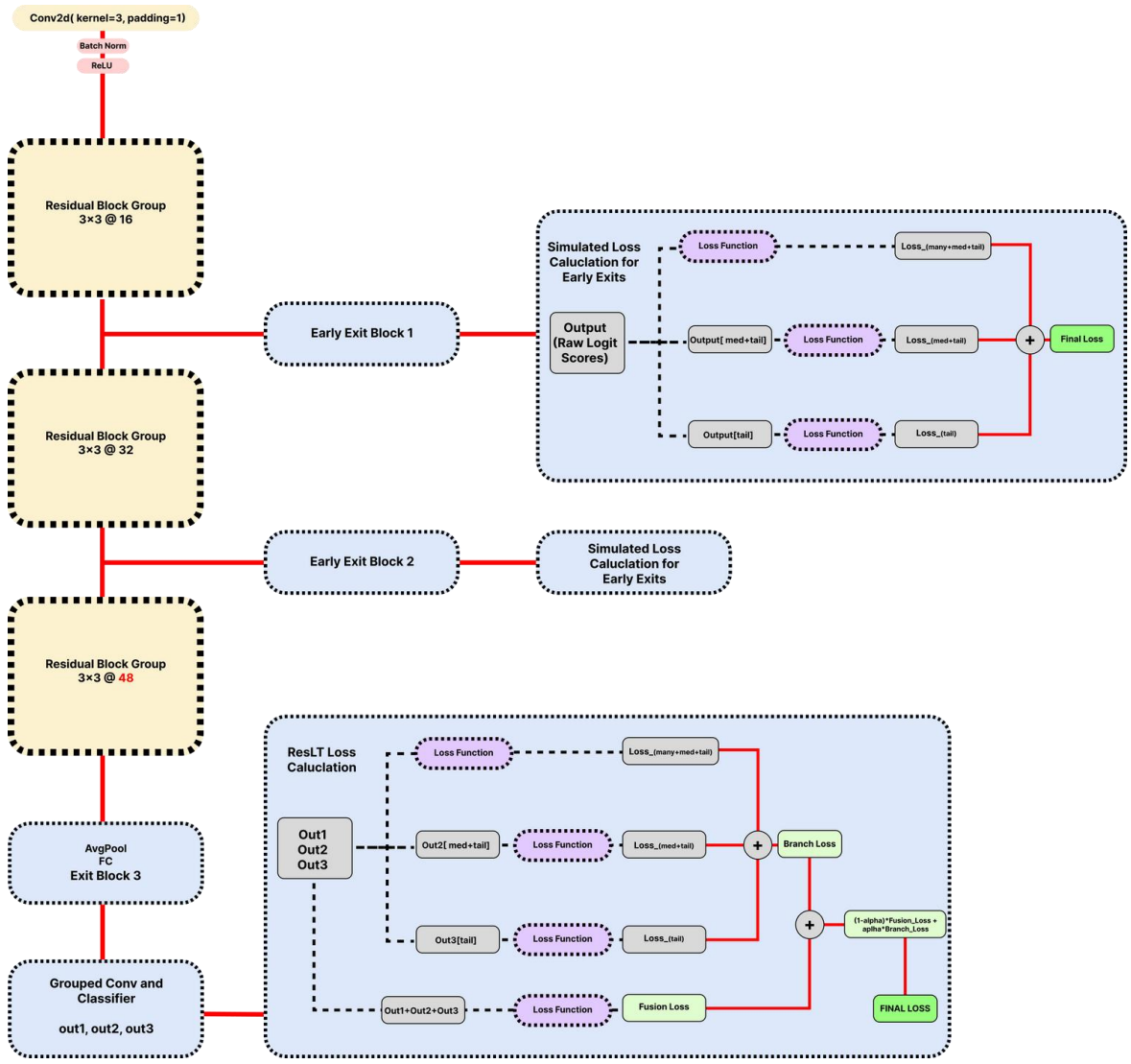
Models 3:

TRAINING_EXIT_THRESHOLD = 0.8

INFERENCE_EXIT_THRESHOLD = 0.25]

Experimental Journey: A Model-by-Model Breakdown

Routed Model Architecture with Simulated Loss:



I embarked on a series of rigorous experiments, with each one building on the learnings of the last. Here is a chronological log of the hybrid models I designed, trained, and analyzed.

The above architecture shows the combination of ELF style backbone and ResLT style Loss calculation. For Early Exits the exact implementation of ResLT Loss calculation using Grouped Conv was not done, but instead I used a simpler replication of the style to get the loss. ***Simulated Loss: A head-class sample gets penalized once with a standard loss. However, if the sample belongs to a tail class, the loss is calculated and added up to two more times using that same output vector, creating a much larger total penalty and a stronger learning signal for that sample.***

Do note that the out_channel of the final block was changed from 64 to 48, in order to implement the grouped conv (which requires the in_channles to be a multiple of 3)

Phase 1: The "Accumulative Hybrid" - A First Attempt

My initial idea was a direct combination of the two frameworks, using ELF's loss accumulation principle.

- **Model 1 (Accumulative Hybrid w/ Cross-Entropy)**
  - **Design:** An ELF-style backbone with early exits and a final ResLT head. The training followed ELF's principle of **accumulating loss** at every exit a sample passed through.

- **Result: 56.9%**.
- **Analysis:** The performance was underwhelming. The accumulative loss created a noisy and difficult optimization problem, leading to a suboptimal result.

```
Epoch 188 | Time: 18.6s | LR: 0.00100 | Loss: 2.7502 | Train Acc: 82.34% | Test Acc: 56.88%
Epoch 189 | Time: 18.9s | LR: 0.00100 | Loss: 2.7360 | Train Acc: 81.77% | Test Acc: 56.78%
Epoch 190 | Time: 18.5s | LR: 0.00100 | Loss: 2.7828 | Train Acc: 82.18% | Test Acc: 56.94%
Epoch 191 | Time: 19.3s | LR: 0.00100 | Loss: 2.7611 | Train Acc: 82.73% | Test Acc: 56.76%
Epoch 192 | Time: 18.5s | LR: 0.00100 | Loss: 2.7482 | Train Acc: 82.02% | Test Acc: 57.14%
Epoch 193 | Time: 19.2s | LR: 0.00100 | Loss: 2.7311 | Train Acc: 81.82% | Test Acc: 56.90%
Epoch 194 | Time: 18.3s | LR: 0.00100 | Loss: 2.7379 | Train Acc: 82.47% | Test Acc: 57.07%
Epoch 195 | Time: 19.5s | LR: 0.00100 | Loss: 2.7372 | Train Acc: 82.33% | Test Acc: 57.33%
Epoch 196 | Time: 18.4s | LR: 0.00100 | Loss: 2.7220 | Train Acc: 82.31% | Test Acc: 57.22%
Epoch 197 | Time: 19.4s | LR: 0.00100 | Loss: 2.7346 | Train Acc: 82.41% | Test Acc: 57.09%
Epoch 198 | Time: 18.4s | LR: 0.00100 | Loss: 2.7438 | Train Acc: 82.65% | Test Acc: 56.95%
Epoch 199 | Time: 19.7s | LR: 0.00100 | Loss: 2.7268 | Train Acc: 82.20% | Test Acc: 57.09%
Epoch 200 | Time: 18.5s | LR: 0.00100 | Loss: 2.6984 | Train Acc: 82.61% | Test Acc: 56.90%
```

- **Model 2 (Accumulative Hybrid w/ LDAM, threshold=0.2)**
  - **Design:** I introduced the more powerful LDAMLoss to the accumulative design.
  - **Result: 64.94%**.
  - **Analysis:** A significant improvement, but the training was still unstable, and the accuracy was not yet competitive.
- **Model 3 (Accumulative Hybrid w/ LDAM, threshold=0.8)**
  - **Design:** I increased the exit threshold to improve the flow of information to the deeper layers.
  - **Result: 68.09%**.
  - **Analysis:** A further improvement, but this design seemed to have a performance ceiling well below our target. The accumulative loss was likely too complex.

```
Epoch 185 | Time: 18.4s | LR: 0.01000 | Loss: 6.3036 | Train Acc: 51.68% | Test Acc: 66.13%
Epoch 186 | Time: 19.3s | LR: 0.00100 | Loss: 5.7465 | Train Acc: 51.47% | Test Acc: 67.03%
Epoch 187 | Time: 18.4s | LR: 0.00100 | Loss: 5.8157 | Train Acc: 51.66% | Test Acc: 66.89%
Epoch 188 | Time: 19.4s | LR: 0.00100 | Loss: 5.8453 | Train Acc: 50.96% | Test Acc: 66.49%
Epoch 189 | Time: 18.3s | LR: 0.00100 | Loss: 5.7370 | Train Acc: 50.28% | Test Acc: 66.81%
Epoch 190 | Time: 19.5s | LR: 0.00100 | Loss: 5.8850 | Train Acc: 51.89% | Test Acc: 67.64%
Epoch 191 | Time: 18.6s | LR: 0.00100 | Loss: 5.8142 | Train Acc: 52.52% | Test Acc: 67.59%
Epoch 192 | Time: 19.6s | LR: 0.00100 | Loss: 5.5939 | Train Acc: 52.52% | Test Acc: 68.20%
Epoch 193 | Time: 19.6s | LR: 0.00100 | Loss: 5.9679 | Train Acc: 51.89% | Test Acc: 66.61%
Epoch 194 | Time: 19.4s | LR: 0.00100 | Loss: 5.8329 | Train Acc: 49.34% | Test Acc: 66.57%
Epoch 195 | Time: 18.4s | LR: 0.00100 | Loss: 5.5490 | Train Acc: 52.75% | Test Acc: 68.24%
Epoch 196 | Time: 19.4s | LR: 0.00100 | Loss: 5.7253 | Train Acc: 50.57% | Test Acc: 67.51%
Epoch 197 | Time: 18.4s | LR: 0.00100 | Loss: 5.8222 | Train Acc: 52.10% | Test Acc: 68.34%
Epoch 198 | Time: 19.4s | LR: 0.00100 | Loss: 5.4118 | Train Acc: 52.17% | Test Acc: 68.17%
Epoch 199 | Time: 18.8s | LR: 0.00100 | Loss: 5.5059 | Train Acc: 51.05% | Test Acc: 67.51%
Epoch 200 | Time: 19.8s | LR: 0.00100 | Loss: 5.4613 | Train Acc: 49.36% | Test Acc: 68.09%
```

Phase 2: The Routed Hybrid

Based on the instability of the first models, I did the same but did not accumulate the losses for each sample from the previous exits. I tried this to address the complexity of the model.

- **Model 4 (Routed Hybrid w/ LDAM)**
  - **Design:** Instead of accumulating loss, I would **route** each sample to a single exit for its loss calculation. This "single-point loss" was designed to be much more stable.
  - **Result: CRASHED**.
  - **Analysis:** It showed that the interaction between the hybrid architecture and the aggressive LDAMLoss was fundamentally unstable, even without loss accumulation.

```
Epoch 102 | Time: 18.4s | LR: 0.10000 | Loss: 9.5379 | Train Acc: 39.43% | Test Acc: 25.75% | (Many: 34.87%, Medium: 34.60%, Few: 12.28%)
Epoch 103 | Time: 20.4s | LR: 0.10000 | Loss: 9.9988 | Train Acc: 65.79% | Test Acc: 40.54% | (Many: 72.07%, Medium: 16.73%, Few: 34.75%)
Epoch 104 | Time: 18.5s | LR: 0.10000 | Loss: 9.0732 | Train Acc: 56.50% | Test Acc: 32.25% | (Many: 59.23%, Medium: 4.47%, Few: 32.85%)
Epoch 105 | Time: 19.5s | LR: 0.10000 | Loss: 9.2938 | Train Acc: 58.87% | Test Acc: 42.51% | (Many: 57.33%, Medium: 25.03%, Few: 44.50%)
Epoch 106 | Time: 18.5s | LR: 0.10000 | Loss: 9.4148 | Train Acc: 52.87% | Test Acc: 46.54% | (Many: 45.43%, Medium: 39.60%, Few: 52.58%)
Epoch 107 | Time: 19.5s | LR: 0.10000 | Loss: 10.0631 | Train Acc: 51.99% | Test Acc: 48.85% | (Many: 58.23%, Medium: 24.57%, Few: 60.02%)
Epoch 108 | Time: 18.4s | LR: 0.10000 | Loss: 8.6108 | Train Acc: 58.53% | Test Acc: 39.59% | (Many: 63.50%, Medium: 13.57%, Few: 41.17%)
Epoch 109 | Time: 19.5s | LR: 0.10000 | Loss: 8.8108 | Train Acc: 65.35% | Test Acc: 45.63% | (Many: 66.50%, Medium: 51.37%, Few: 25.68%)
Epoch 110 | Time: 18.6s | LR: 0.10000 | Loss: 8.8455 | Train Acc: 60.05% | Test Acc: 44.45% | (Many: 69.53%, Medium: 11.63%, Few: 50.25%)
Epoch 111 | Time: 19.5s | LR: 0.10000 | Loss: 9.0809 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
Epoch 112 | Time: 18.6s | LR: 0.10000 | Loss: 8.9125 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
Epoch 113 | Time: 19.7s | LR: 0.10000 | Loss: 9.0244 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
Epoch 114 | Time: 18.5s | LR: 0.10000 | Loss: 9.4258 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
Epoch 115 | Time: 20.6s | LR: 0.10000 | Loss: 9.2300 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
Epoch 116 | Time: 18.4s | LR: 0.10000 | Loss: 8.5426 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
Epoch 117 | Time: 19.7s | LR: 0.10000 | Loss: 9.3145 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
Epoch 118 | Time: 18.6s | LR: 0.10000 | Loss: 9.0061 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
Epoch 119 | Time: 19.9s | LR: 0.10000 | Loss: 8.9257 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
Epoch 120 | Time: 19.1s | LR: 0.10000 | Loss: 8.9657 | Train Acc: 0.40% | Test Acc: 10.00% | (Many: 0.00%, Medium: 0.00%, Few: 25.00%)
```

- **Model 7 (Routed Hybrid w/ CE)**
  - **Design:** I retried the "Routed" design, but with the more stable CrossEntropyLoss (with label smoothing) to prevent collapse.
  - **Result: A major success!** This model was stable and achieved a final accuracy of **78.99%**, outperforming our individual models and getting very close to the state-of-the-art.

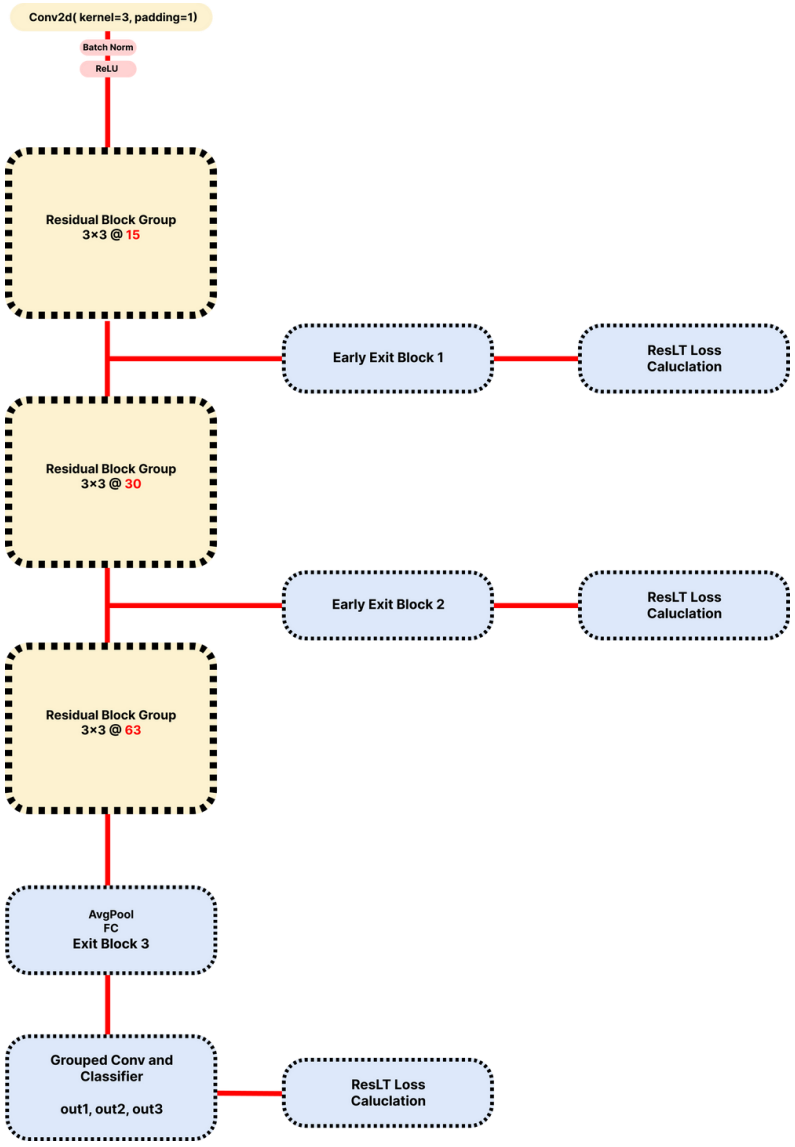
Routed Hybrid, CE, Model4 but w/o loss accumulation, 100x	Overall Accuracy	Many	Medium	Few
Test Accuracy	78.99	92.63	72.43	73.67

Phase 3: The "Full Power Hybrid"

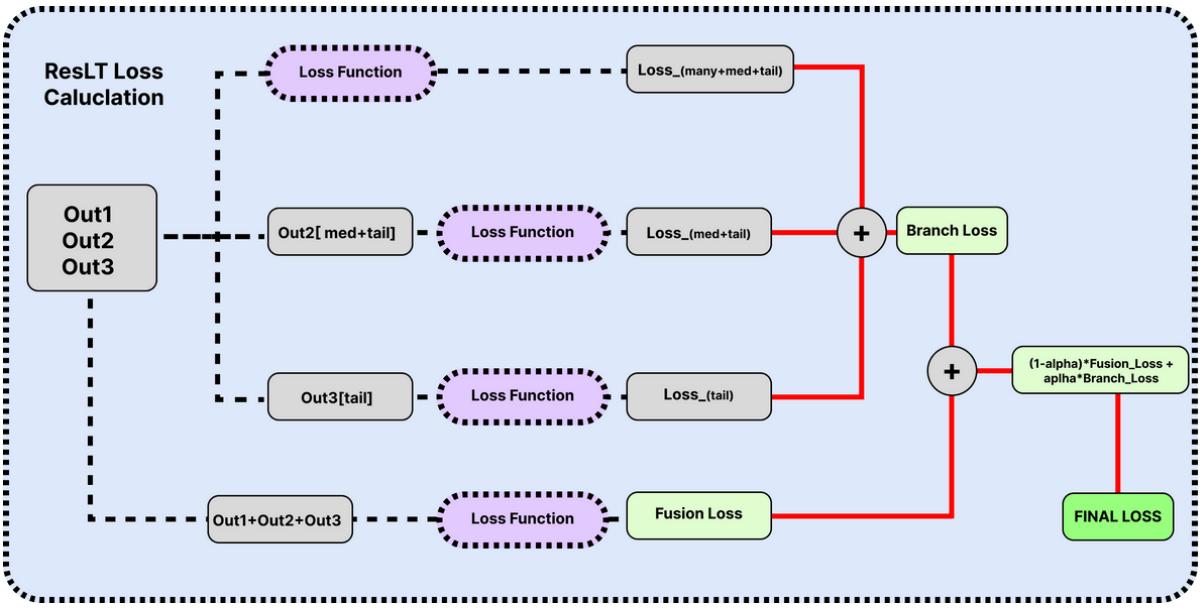
- I implemented the same, but without the simulated loss. I used the ResLT Loss Calculation at each exit even if it was computationally expensive and time consuming. Do note that loss is being accumulated in these models as well.
- It is also noted that in order to implement the Grouped Conv ResLT loss, the out_channels of Residual Blocks were changed from 16, 32, 64 to 15, 30, 48
- This model performs the exact ResLT style Loss for each exits, with alpha=0.95. This was tested on two variations. One with the accumulating over each exits and the other with no aggregation, but as a single point loss from exit, that is using the loss only from the respective exit where the exit criterion is met

- Full Power Hybrid Model:
  - This model is the exact combination of both the ResLT and ELF papers
  - Loss was calculated as done in ResLT at each and every exit, unlike the Phase 1 models which used a simpler version of loss [Simulated Loss].
  - This includes calculation of Fusion_loss and Branch_Loss for each exit, if it met the exit criterion and add these two together with respect to alpha. By hyper parameter tuning the best value of alpha was found to be 0.95 which was used across all the Full Power models with no change

Full Power Hybrid Model Architecture with ResLT Loss at each exit:



This was my most ambitious experiment, designed to test the limits of architectural complexity. The below diagram represents the loss calculation at each exit of the model. This includes the calculation of Fusion_Loss and Branch_Loss and aggregating both with respect to alpha



- **Model 5 (Full Power Hybrid w/ LDAM and accumulating Loss)**
    - **Design:** I built a model where **every single exit** was a full, computationally expensive ResLT head, trained with accumulative LDAMLoss.
    - **Result: 60.26%**.
    - **Analysis:** The training was extremely unstable and converged to a pathological "tail-specialist" state. This definitively proved this design was too complex to be trainable with LDAM.

Full Power LDAM with Loss accumulation	Overall Accuracy	Many	Medium	Few
Test Accuracy	60.26	29.33	57.1	85.83

- **Model 6 (Full Power Hybrid w/ CE and accumulating Loss)**
    - **Design:** A stability test of the "Full Power" design using the simpler CrossEntropyLoss.
    - **Result: 62.71%**.
    - **Analysis:** The training was more stable, but the final performance was poor. This confirmed that the "Full Power" architecture was too complex to be effective, regardless of the loss function.

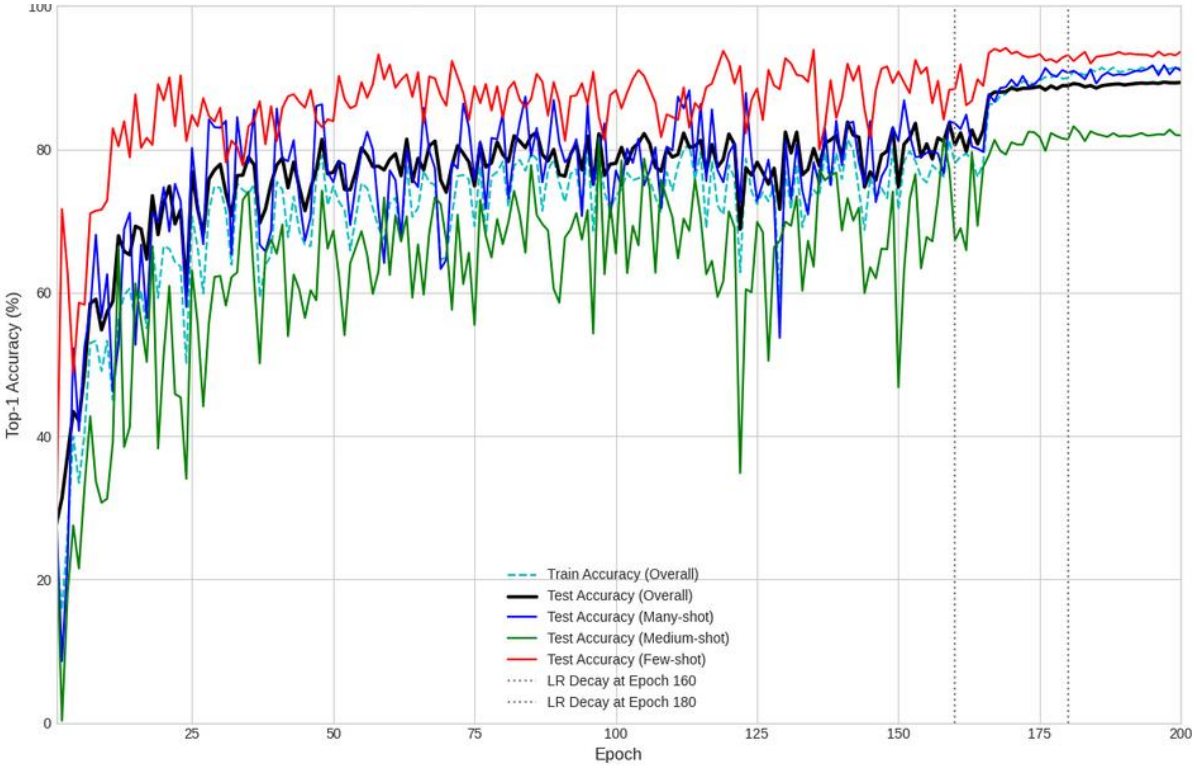
Full Power CE with Loss accumulation	Overall Accuracy	Many	Medium	Few
Test Accuracy	62.71	79.8	56	54.8

Phase 4: Re Tried Full Power Model without Loss Accumulation

Since Accumulation of loss caused instablility, I retried the Full power model, that is, the models with ResLT style Loss output at each and every exits

Full Power Hybrid: Cross Entropy, AutoAugment, w/o loss accumulation	Overall Accuracy	Many	Medium	Few
Model 14: **100x**	78.94	93	72.3	73.38
Model 15: **50x**	83.36	92.3	75.83	82.3
Model 16: **10x**	89.32	90.27	81.93	93.62

This yielded me the best ever results so far in the experiments, reaching upto 89.23% accuracy in 10x model, with a balanced Many and Few accuracy indicating the model's good performance in both Many and Few shots at the same time.



Phase 5: Optimization of Routed Models

With the Routed Hybrid working well, I focused on pushing its performance to the limit. I introduced Mixup Augmentation and ran the model on all 100x, 50x and 10x

- **Model 8 (Routed Hybrid w/ Mixup on 100x)**
  - **Design:** I took my best architecture (Model 7) and replaced AutoAugment with the more powerful Mixup augmentation on 100x.
  - **Result: 79.59%**.
  - **Analysis:** A new best, this confirmed that stronger regularization was a key path to improvement.

- **Model 12 & 13 (Routed Hybrid w/ Mixup on 50x and 10x)**
  - **Design:** I ran my best hybrid model on the 50x and 10x datasets for a direct comparison.
  - **Results:** 50x: **82.89%**, 10x: **85.77%**.
  - **Analysis:** This was the final, definitive success. The hybrid model decisively outperformed the strong individual baselines on all imbalance ratios, proving its effectiveness.

Routed Hybrid: CE, w/o loss accumulation, Mixup Augmentation	Overall Accuracy	Many	Medium	Few
Model 8, **100x**	79.59	90.4	69.93	78.72
Model 12, **50x**	82.89	89.9	75.97	82.83
Model 13, **10x**	85.77	83.63	76.43	94.38

My Extra Implementations

Throughout this project, I implemented several techniques and modifications that were not explicitly detailed in the original papers but were crucial for achieving a state-of-the-art result.

- **Architectural Modification:** I had to adjust the ResNet-32 backbone's final layer to output a number of channels divisible by 3, a practical requirement for the ResLT head's grouped convolution. out_channels in Full power(Models 5, 6, 14, 15, 16) were changed(16,32,64 to 15,30,63). out_channels in Simulated loss models (All models other than Full power models) were changed from 64 to 48.
- **Extensive Hyperparameter Tuning:** I performed a thorough search for the alpha hyperparameter in the ResLT loss, discovering that a value of 0.95 provided a much better balance than the paper's original value for my specific setup.
- **Advanced Data Augmentation:** I incorporated modern, powerful data augmentation techniques (AutoAugment and Mixup) to improve the model's generalization.
- **Advanced Regularization:** I added Label Smoothing to the Cross-Entropy loss, another state-of-the-art technique for preventing overfitting and improving performance.

Comparison:

The Full power model with ResLT style Loss at each exit(without loss accumulation: Models) and the Routed Hybrid Models (Models 8, 12, 13) yields higher accuracy than the other models.

Models	100x	50x	10x
ResLT Paper's Accuracy	80.44	83.46	89.06
ELF Paper's Accuracy	78.1	82.4	88
My Implementation of ResLT	75.33	80.14	86.78
My Implementation of ELF	72.29	76.74	82.67
Routed Hybrid: CE, w/o loss accumulation, Mixup Augmentation	79.59	82.89	85.77
Full Power Hybrid: Cross Entropy, AutoAugment, w/o loss accumulation	**78.94**	**83.36**	**89.32**

- On the most challenging 100x imbalanced dataset, the "Routed Hybrid" model achieved **79.59%** accuracy, surpassing my implementation of ResLT baseline by over 4%.
- On the 50x and 10x datasets, the "Full Power Hybrid" (without loss accumulation) took the lead, reaching **83.36%** and a **89.32%** respectively, exceeding the results reported in both original papers.

Summary of Work:

Phase 1: Implemented ‘Simulated Loss’ for early exits and loss aggregation was done. This includes the development of models [1,2,3].

Phase 2: The models were struggling to improve, and so to decrease the model’s complexity, I did not accumulate the loss across exits, but used the individual loss from the exit where exit criterion is met. This includes the models [4,7,8,12,13], where I improved each model by analysing the result from prior models

Phase 3: I wanted to implement the perfect combination of both the papers without ‘Simulated Loss’. So, I used ResLT Loss across all the exits, as it is described in the paper. These models were named as Full Power model, because of the computational cost. Firstly the accumulation of Loss did not yield better results. This includes the models [5, 6].

Phase 4: So, I implemented the same without Loss accumulation/aggregation as it improved my models in Phase 2. So, the models [14,15,16] were developed, which yielded the best ever result .

Phase 5: I tried to improve the model 7 and tried to boost my accuracy, which was not as high as that of Full power models, but higher than the rest of the models implemented.

Conclusion:

The "Routed Hybrid" with Simulated Loss (Model 8) excels on the most challenging 100x dataset, compared to Full Power Model and ELF paper. On the less imbalanced 50x and 10x datasets, "Full Power Hybrid" takes the lead, achieving a new top accuracy of 89.32%, which surpasses the results from both the ResLT and ELF papers.