ChatGPT

# Final CIFAR-10 Experiment: ResNet-Style Model for ≥91% Accuracy

## Current Model Bottlenecks and Training Dynamics

The existing **m6** model (6 residual blocks) has ~**699k** parameters [1] and reached ~89% validation accuracy while nearly **100% train accuracy** [2]. This large generalization gap indicates severe overfitting – the network memorized training data but struggled to generalize. Its **wide layers** (32→64→128 filters) provide high capacity, and indeed the train loss plunged to near zero, yet validation accuracy plateaued below 0.90. Even with learning-rate drops, m6 peaked around **89.1% val** accuracy [3]. This suggests the current architecture, though high in parameters, isn't efficiently using its capacity to push beyond 90% test accuracy. In short, **insufficient depth** and suboptimal regularization are bottlenecking performance.

## Proposed Architecture Modifications (Deeper & Slimmer ResNet)

To boost accuracy, we recommend a **deeper ResNet-style model** with more residual blocks but **fewer filters per layer**. This design increases representational power via depth (mitigating underfitting) while reducing total parameters (mitigating overfitting):

- **Increase Depth:** Expand from 6 to **9 residual blocks** (a ResNet-20 architecture) across three stages [4]. Deeper networks can learn more complex features and have proven gains in accuracy – e.g. ResNet-32 (0.46M params) outperforms ResNet-20 (0.27M) by ~1.2% [5]. Extra blocks address the under-utilization of capacity observed in m6.
- **Reduce Width:** Use **16→32→64 filters** (instead of 32→64→128) in the three stages [6]. This cuts parameters by ~70–75% with only ~1–2% potential accuracy drop [6]. The parameter budget falls to ~**270k**, well below 400k, which greatly lowers overfitting risk. Notably, such a slimmer ResNet still retains high accuracy – many works report ~89–91% with ≤400k params given proper training [7].

**Proposed ResNet-20 Architecture (CIFAR-10):**

- **Input:** 32×32 RGB image
- **Conv1:** 3×3 conv, 16 filters, stride 1 → BatchNorm → ReLU
- **Stage 1 – 3 Residual Blocks @ 16 filters:** For each block: Conv 3×3 → BN → ReLU → Conv 3×3 → BN → **Add** (skip connection) → ReLU. (All convs use 16 filters; identity shortcuts since input=output channels.)
- **Stage 2 – 3 Residual Blocks @ 32 filters:** First block uses Conv 3×3, 32 filters, **stride 2** (downsample) with a parallel 1×1 conv skip (16→32) [8]. This block: Conv(3×3, s=2) → BN → ReLU → Conv(3×3, s=1) → BN → Add(projection skip) → ReLU. The next two blocks: 3×3 convs with 32 filters (stride 1) and identity skips.
- **Stage 3 – 3 Residual Blocks @ 64 filters:** First block: Conv(3×3, 64f, s=2) with 1×1 skip (32→64) to downsample, as above. Then two more blocks of 3×3 convs at 64 filters with identity shortcuts.
- **Output:** Global Average Pooling → Dropout(**0.3**) → Dense(10 classes, softmax).

This ResNet-20–style network preserves the beneficial **skip connections** of ResNet (no added cost in params) which ensure training stability even at greater depth [4]. Despite a smaller size, it should achieve our target accuracy: **ResNet-20 (16/32/64 filters)** historically reaches ~**91–92%** on CIFAR-10 with augmentation [9]. By comparison, our previous m6 was wider but shallower – the new model is more parameter-efficient and aligned with architectures known to exceed 91% accuracy. *(If desired, we could allocate more filters (e.g. 20→40→80, ~427k params) up to the 600k ceiling for a slight buffer in capacity, but experiments suggest it may not be necessary [7].)*

## Training Configuration and Hyperparameters

We will also refine the training strategy to maximize generalization from this new architecture. Key settings are chosen based on ResNet best practices and our past experiments:

```
// Training config for final CIFAR-10 run
{
  "optimizer": "SGD",
  "momentum": 0.9,
  "learning_rate": 0.05,
  "lr_schedule": "cosine_annealing_150ep",
  "weight_decay": 5e-4,
  "dropout_rate": 0.3,
  "batch_size": 32,
  "epochs": 150,
  "data_augmentation": {
    "random_crop": true, "crop_padding": 4,
    "horizontal_flip": true,
    "mixup_alpha": 0.2
  },
  "early_stopping": false
}
```

- **Optimizer & LR Schedule:** Use stochastic gradient descent (**SGD** + 0.9 momentum) as in prior runs, as it tends to yield better final generalization than Adam for CNNs given sufficient epochs [10]. We start with a fairly high **learning rate (0.05)** (BatchNorm allows even 0.05–0.1 [11]) and employ a **cosine annealing** schedule over **150 epochs** [12]. This gradually decays the LR from 0.05 to ~0 by epoch 150, enabling small, fine weight updates in later training. The cosine schedule ensures the model doesn't "stall" – even if validation improvement pauses, the continuously decreasing LR will coax out further accuracy gains. (Alternatively, a step decay at epoch 80 and 120 could be used, but cosine is smooth and was suggested in earlier tuning [13].)
- **Regularization (L2 & Dropout):** Keep **weight decay = 5×10^−4** on conv and dense weights, as used before, to penalize large weights (this value worked well in prior CIFAR models and in ResNet papers [14]). **Dropout 0.3** after the global pooling layer is retained to combat overfitting; this moderate rate adds regularization without crippling learning. Since the new model has far fewer parameters, we avoid over-regularizing (e.g. we do **not** raise dropout to 0.5, which earlier analysis found could hurt smaller models [15]). The chosen 0.3–0.4 range for dropout + 5e-4 L2 has proven effective for mid-sized models like this [16].

- **Data Augmentation:** Enable **on-the-fly augmentation** to expand effective training data. We will use standard CIFAR-10 techniques: **random horizontal flips** and **random crops** of size 32×32 (after padding images by ~4 pixels) [17]. This introduces new image variants (shifts, reflections) each epoch, which consistently boosts generalization in our experiments [18]. In addition, we propose using **MixUp** (with α≈0.2), which linearly blends pairs of training images and labels. MixUp further regularizes the model by requiring linear behavior between classes and has been shown to improve CIFAR-10 accuracy by ~1–2% in many cases. These augmentation strategies will help the network resist overfitting and reach higher test accuracy [19]. *(If MixUp is not already supported in our pipeline, it can be omitted, but it is a one-time addition that prior configs earmarked for improved generalization [20].)*
- **Epochs & Early Stopping:** We extend training to **~150 epochs** to give the deeper model ample time to converge. Past ResNet trials (e.g. He et al.) often use 164–200 epochs on CIFAR-10 [12] – our schedule should similarly reach peak performance near the tail end of training. Early stopping will be **disabled (or patience set very high)** because the validation accuracy may plateau for many epochs then improve once the learning rate is sufficiently reduced. For example, in m6 the val accuracy stalled in the 0.88–0.89 range until after epoch 80 when LR drops triggered a jump to ~0.891 [21]. We expect a similar late-phase uptick in this run, so the model should be allowed to train to completion. We will monitor validation metrics each epoch and only stop early if we observe a clear decline even at low learning rates.

## Expected Impact and Validation Metrics

- **Model Capacity & Parameters:** The modified network has ~**270k** trainable parameters (plus ~1.8k in BN stats) – roughly **1/4** of m6's size [1]. This drastic reduction, via narrower layers, directly addresses the overfitting issue. The smaller model is less likely to memorize training data outright, especially when coupled with strong regularization. Yet thanks to added depth and residual learning, it should maintain **high representational power**. (Identity skips cost no extra parameters, so increasing depth is a "free" way to gain accuracy [22].) In effect, we are trading excess width for useful depth – a strategy known to improve CIFAR-10 accuracy per parameter [7]. We anticipate only a minor accuracy trade-off for staying under 400k params, well worth the generalization benefits [6].
- **Accuracy Goal ≥ 91%:** This experiment is designed to surpass **91% test accuracy**. Based on published results and our fine-tuning research, a ResNet-20 with 16–64 filters can achieve around **91–92%** on CIFAR-10 **with augmentation** [9] [7]. In fact, the original ResNet-20 (which is what we're implementing) obtained ~**8.75% test error** without aug (≈91.3% accuracy) [9]. With our use of flips/crops (and possibly MixUp), we expect to beat that. For reference, Hideyuki Inada's Keras CIFAR-10 experiment (with a similar ResNet approach, plus heavy aug and LR scheduling) reached **91.93%** test accuracy [23]. Our configuration mirrors those successful strategies, so achieving **91–92%** is realistic. Exceeding 93% may be possible if all goes well, but our primary objective is to confidently clear the 91% bar.
- **Training Curve and Generalization:** We should prepare to observe a different training dynamic with the new setup. Early in training, the **train accuracy** may still climb rapidly (though perhaps not to 99.9% as before), while **validation accuracy** will start lower (due to augmented, harder data) and improve gradually. It's normal if the model still reaches ~100% train accuracy eventually – the key is that validation will now track upward more steadily instead of flat-lining in the 80s. By mid-training (epoch ~75), we expect val accuracy in the high-80s. In the final third of training (epochs ~100–150), as the learning rate enters the **10^−3 to 10^−4** range, the model should fine-tune its filters and push validation accuracy into the 0.90+ territory. We will **monitor** the validation curve for an upward

trend: small improvements of +0.1–0.3% per epoch in the 90% range are signs that the cosine LR schedule is doing its job. If the validation accuracy plateaus long even as training accuracy continues rising, that would indicate overfitting – in such case, we might increase the mixup strength or dropout next time. However, given the current plan, we expect to see the **generalization gap shrink** significantly (e.g. final train acc ~99%, val acc ~91%+, gap ~8% vs. 11% before). The combination of more data (augmented) and more constrained model size should yield a much tighter train/test performance. [2] [7]

In summary, this final experiment leverages a **deeper, slimmer ResNet architecture** alongside aggressive regularization (data augmentation, weight decay, dropout) and a proven LR schedule. These changes directly target m6's issues by improving effective capacity and reducing overfit. We expect the new model to attain the **≥91% CIFAR-10 test accuracy** goal while respecting the ~600k parameter budget – and ideally, to do so with a comfortable margin [7], as evidenced by similar ResNet results in the literature. Each component of this plan (ResNet depth, CIFAR-10 augmentation, cosine LR decay) has been validated in research and prior runs, so together they maximize our chances of success in this one-shot training run. **91%+ accuracy** is within reach, and we will closely track the training/validation curves to ensure the model is on the right trajectory towards this target. [23] [7]

---

[1] [2] [3] [21] log.txt
file://file-5kJ1fmCKYBGaSrANT9JUwy

[4] [10] [11] [19] [22] t001_fine-tuning-01.pdf
file://file-41ZNDsVMAWknDrxKYpHweA

[5] [9] [14] t001_1512.03385v1.pdf
file://file-LTcXMbN36yAX1RrG8ypqpP

[6] [7] t001_fine-tuning-05.txt
file://file-LZtFMX5YqiAMtasVVdaVgi

[8] model.py
file://file-StoSooUaBDCa9zMppYN6st

[12] [17] [23] t001_fine-tuning-04.txt
file://file-6kUsLRcN2tRe7DeBLVSmFR

[13] [15] [16] [18] [20] t001_fine-tuning-03.txt
file://file-NHWjVoRw9F3rbRohJJQLA4