Transfer Learning Optimization Investigating normalization techniques and gradient dynamics in MobileNetV2 on CIFAR-10

Hiva Abolhadizadeh
Special Topics in Artificial Intelligence
Supervisor: Dr. M. Eftekhari

September 25, 2025

Abstract

In this study, the effect of three normalization methods, including BatchNorm, LayerNorm, and FRN, is investigated in the context of fine-tuning pre-trained models. We analyze how these approaches influence training stability and convergence, and we evaluate the role of gradient clipping as an additional stabilization mechanism. The experiments are performed by fine-tuning MobileNetV2 on CIFAR-10. Results indicate that the choice of normalization at the head of the network significantly affects both convergence speed and final accuracy.

Contents

1 Introduction

Transfer learning is a key paradigm in modern machine learning that enables models to leverage knowledge acquired from solving one task to address a different task. This approach is particularly effective in scenarios with limited training data. Instead of training a model from scratch, a pre-trained model is adapted and optimized for the target task. This process, commonly referred to as fine-tuning, has been extensively applied in recent years across computer vision, natural language processing, and other domains.

In training deep models, one of the decisive factors influencing stability and convergence speed is the normalization of data and network layers. Techniques such as Batch Normalization, Layer Normalization, and Filter Response Normalization (FRN) adjust feature statistics during training, thereby enabling models to learn more efficiently and robustly. However, the behavior of these normalization methods within the context of transfer learning—particularly when adapting pre-trained models to novel datasets—remains insufficiently explored and warrants further investigation.

The objective of this project is to conduct an empirical and systematic evaluation of various normalization strategies in transfer learning. To this end, we employ MobileNetV2 as the base architecture, a lightweight and efficient model designed for low-resource devices and originally trained on large-scale datasets such as ImageNet. The dataset used in this study is CIFAR-10, comprising 60,000 color images across 10 distinct classes, each with a resolution of 32×32 pixels. For compatibility with MobileNetV2, all images were resampled to 224×224.

Within this study, three alternative normalization heads were designed and integrated into the base model. Furthermore, we investigated the effect of Gradient Clipping to assess its role in enhancing training stability and shaping gradient dynamics during fine-tuning. For a comprehensive analysis, we employed diagnostic tools such as loss landscape analysis and gradient monitoring.

The central research questions addressed in this project are as follows:

Which type of normalization yields superior performance in transfer learning on CIFAR-10?

Does the application of Gradient Clipping improve training stability?

How do gradient behaviors across different layers of the model vary under distinct normalization conditions?

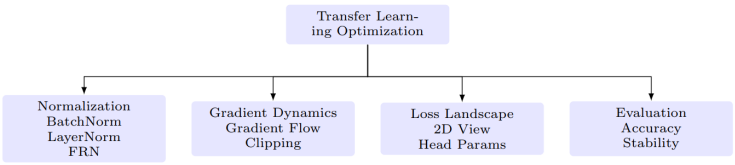

Figure 1: Overview of the components examined in this report

2 Related Work

Transfer learning, particularly in the field of computer vision, has received significant attention in recent years. Leveraging pre-trained models such as ResNet, EfficientNet, and MobileNet for new tasks not only reduces computational costs but also improves model performance. Nevertheless, how these models adapt to novel datasets and the role of normalization techniques during the fine-tuning process remain open and challenging questions in the artificial intelligence literature.

In the study by [Singh and Krishnan, 2020], the authors introduced a novel technique known as Filter Response Normalization (FRN). Unlike Batch Normalization, FRN operates independently of batch size and the statistical distribution of the data. This property makes FRN a particularly suitable option for transfer learning, especially under conditions involving small batch sizes or imbalanced datasets.

Another line of research has highlighted the usefulness of **loss landscape analysis**[1] as a tool for better understanding model behavior during training. The influential work of [Li et al., 2018] demonstrated that the geometry and smoothness of the minima in the loss landscape are directly correlated with a model's generalization ability.

In terms of stabilizing the training of deep networks, Gradient Clipping has been proposed as an effective method to mitigate gradient explosion. Studies such as [Zhang et al., 2019] report that under specific conditions—particularly when fine-tuning deeper layers—clipping can lead to more stable convergence and reduced fluctuations in gradient magnitudes.

Building upon these prior works, the present study seeks to conduct an empirical evaluation of normalization and clipping techniques in the context of transfer learning, focusing on the combination of MobileNetV2 and the CIFAR-10 dataset. To provide deeper insights, we further employ tools such as loss landscape analysis and gradient dispersion monitoring.

[1]Loss Landscape

3 Methodology

This section describes the process of data preparation, the design of the transfer learning model, the implementation of normalization Heads, and the detailed experimental settings. All procedures were implemented in the Google Colab environment using the PyTorch and Torchvision libraries.

3.1 Model Architecture and Modifications

The base model employed in this project is MobileNetV2, pre-trained on the ImageNet dataset and used as a Feature Extractor[2] is one of the most common approaches in transfer learning [Pan and Yang, 2010]. To adapt this model to the CIFAR-10 image classification task, the original classifier was removed, and its output was connected to a newly designed Head responsible for producing the final 10-class output.

Three distinct Head architectures were designed and implemented to investigate the effect of different normalization methods:

- **BatchNormHead**: This Head consists of a Fully Connected (FC) layer with 256 outputs, a Batch Normalization layer[3], and a final output FC layer. The mean and variance are computed across the batch, and normalization is performed as: $\hat{x} = \frac{x - \mu_{\text{batch}}}{\sqrt{\sigma^2_{\text{batch}} + \epsilon}}$, followed by scaling and shifting with parameters (γ, β).
- **LayerNormHead**: This design follows the same structure as BatchNormHead, except that Layer Normalization is applied. Statistics are computed independently of the batch, across features within each sample, and normalization is performed along the internal feature dimension.
- **FRNHead**: This design incorporates Filter Response Normalization (FRN), specifically introduced to avoid dependence on batch statistics. The FRN layer in this head is custom-implemented and combined with a parameter τ: $\max(\gamma x + \beta,\ \tau)$. Unlike the previous two methods, FRN is not dependent on any specific batch or channel.

From a transfer learning perspective, BatchNorm often suffers from degraded performance in cases with small batch sizes[4], due to fluctuating statis-

[2]Using pre-trained models

[3]Batch Normalization plays a critical role in stabilizing gradients and accelerating convergence [Singh and Krishnan, 2020], a ReLU activation function, Dropout

[4]Such as fine-tuning with limited data.

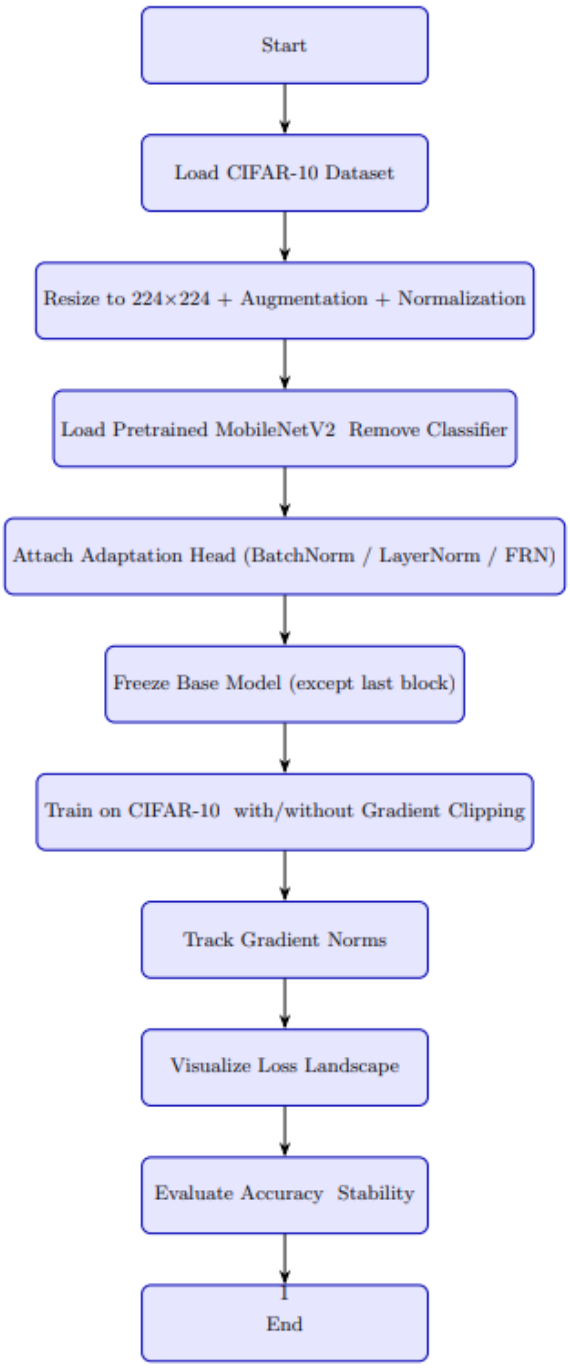

Figure 2: Workflow diagram

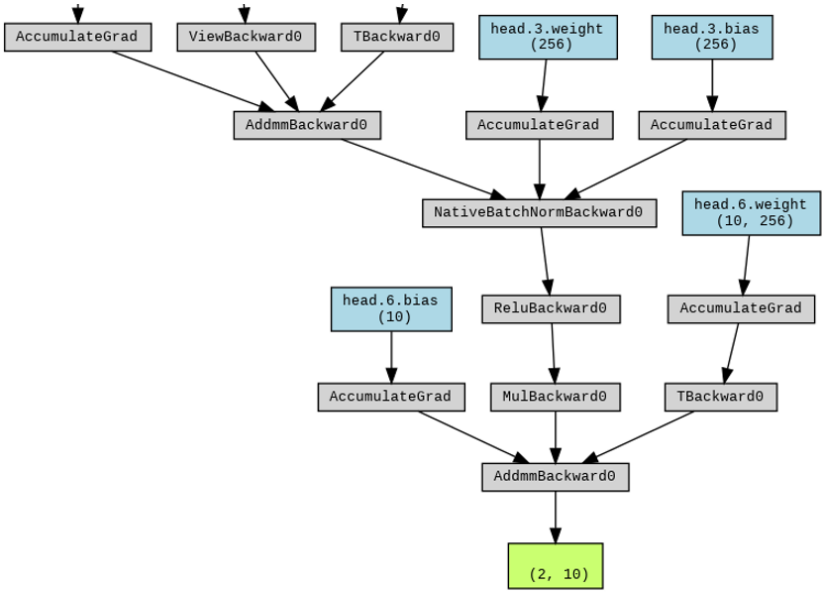

Figure 3: Portion of the model architecture

tics [Singh and Krishnan, 2020]. By contrast, FRN, being independent of such statistics, can achieve better performance in these scenarios. While LayerNorm has been more commonly applied in sequence-based domains such as NLP, it has also shown success in certain Vision Transformer architectures, and is therefore included here for comparative purposes.

In all Heads, the output of the Feature Extractor first undergoes Global Average Pooling, after which the features are flattened into a fixed-size vector and passed into the input FC layer of the head.

To mitigate Overfitting and preserve the general representations learned by MobileNetV2, all base model layers were frozen except for the final block (features 18), which was unfrozen and fine-tuned with a lower learning rate.

Figure 4: BatchNorm Head architecture

Figure 5: LayerNorm Head architecture

Figure 6: FRN Head architecture

3.2 Experimental Setup

The dataset employed in this project is CIFAR-10, which consists of 60,000 color images of size 32×32 across ten classes. To ensure compatibility with the MobileNetV2 input, all images were resampled to 224×224[5].

Figure 7: Examples from the dataset

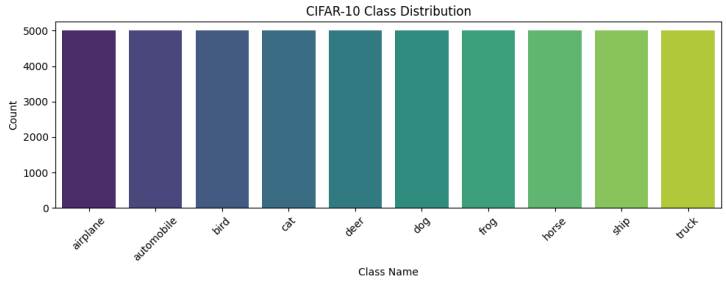

Figure 8: Class-wise data distribution

For the training set, Data Augmentation operations were applied as follows:

- RandomHorizontalFlip
- RandomCrop with padding of 4
- Normalization using the CIFAR-10 mean and standard deviation values:
  $\mu = (0.4914, 0.4822, 0.4465)$
  $\sigma = (0.2023, 0.1994, 0.2010)$

[5]The original input of MobileNetV2 is designed for 224×224 images

For the validation and test sets, only Resize and normalization were applied. The dataset was partitioned into 80% training and 20% validation. A dedicated class named CIFAR10DataLoader was implemented to handle data loading, splitting, and visualization (including class distribution and sample previews).

The experimental design combines three normalization Heads[6] with two conditions: with or without Gradient Clipping (applied with a threshold of 0.1). Consequently, six distinct configurations were evaluated.

The training process employed the CrossEntropyLoss function and the Adam optimizer. The learning rate was set to 10^{-3} for the Head layers and 10^{-5} for the unfrozen layers of MobileNetV2.

Key experimental parameters include:

- Number of epochs: 15
- Batch size: 64 (and 32 for gradient analysis)
- Optimizer: Adam with two distinct learning rates
- Gradient Clipping: applied in half of the experiments with a threshold of 0.1

The training procedure involved monitoring accuracy and loss at each step, saving results, and, when required, performing gradient analysis and loss landscape evaluation for the different models. In addition, a dedicated tool was developed to record per-layer gradients and compare their distributions under clipping and non-clipping conditions [Zhang et al., 2019, Li et al., 2018].

[6]BatchNorm, LayerNorm, FRN

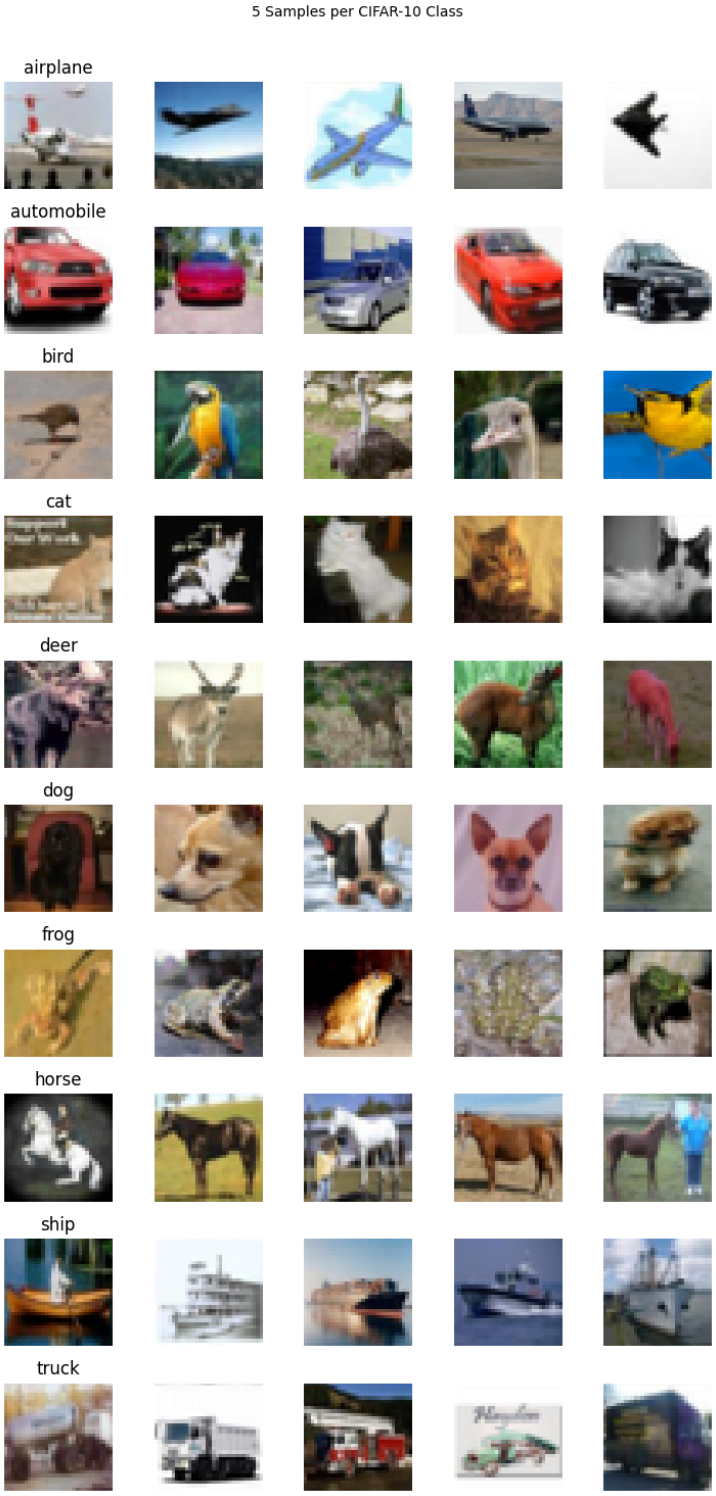

Figure 9: Sample instances from each class

4 Empirical Analysis

4.1 Accuracy and Error

To evaluate the effect of different normalization structures, six separate experiments were conducted by combining three normalization methods[7] with and without the Gradient Clipping technique. All models exhibited a relatively smooth upward trend in validation accuracy over 25 training epochs.

Specifically, the highest final accuracy was achieved by FRN-NoClip with a score of 90.57%. This was followed by BatchNorm-Clip with 90.71%, and LayerNorm-NoClip with 90.22%. The lowest final accuracy was observed in BatchNorm-NoClip with 89.64%.

In terms of convergence, models with Clipping generally demonstrated faster convergence during the early epochs. For instance, the accuracy of BatchNorm-Clip in the first epoch was 75.16%, whereas its non-clipping counterpart registered 73.73%. However, towards later epochs, performance differences diminished, and in some cases, such as with LayerNorm, the effect of Clipping was negligible or even slightly negative.

4.2 Loss Landscape Examination

Despite a technical error in visualizing the loss landscape, the stable convergence of FRN and LayerNorm models without Clipping suggests the presence of smoother loss surfaces in these architectures. Furthermore, the consistent decline in gradient norms over time (from approximately 44 to below 29) further supports this interpretation.

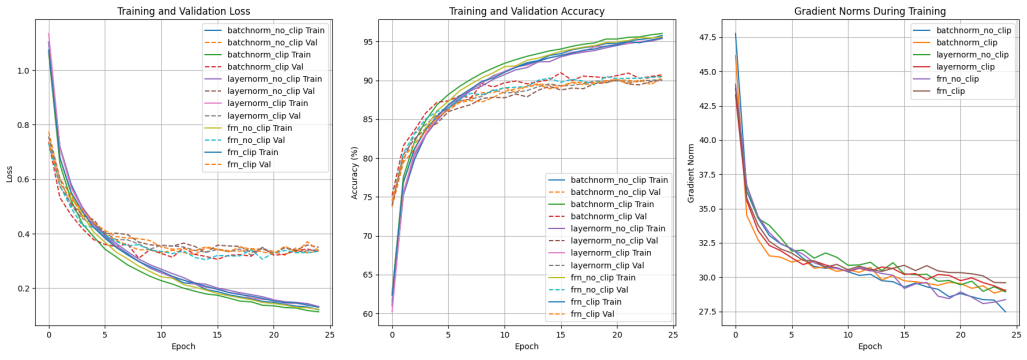

Figure 10: Training analysis of models

[7]BatchNorm, LayerNorm, FRN

5 Discussion

The empirical findings revealed that the choice of normalization method has a direct impact on training stability and performance. The FRN architecture without clipping achieved the highest accuracy, demonstrating inherent stability.

LayerNorm-based models, particularly in the absence of Clipping, also exhibited smooth and stable convergence, which can be attributed to their independence from batch statistics. The evolution of gradient norms in this structure was also notably consistent.

In contrast, BatchNorm without clipping showed greater fluctuations. The application of Clipping improved gradient stability and raised the final accuracy from 89.64% to 90.71%. More specifically, gradient norms in BatchNorm-Clip were better controlled (ranging from approximately 46.15 down to 29.08).

The role of gradient clipping across all structures was either positive (as in BatchNorm) or neutral (as in LayerNorm), and in no case did it result in performance degradation.

6 Conclusion and Future Work

The main findings are summarized as follows:

- The FRN-NoClip structure achieved the best final performance with an accuracy of 90.57%.
- Clipping improved performance in the BatchNorm structure (a gain of +1.07%).
- FRN and LayerNorm achieved stable and accurate performance without the need for Clipping.

Limitations:

- Training was performed only on the model head, with frozen base layers.
- Experiments were conducted under limited computational resources (e.g., Google Colab).
- Loss landscape visualization was not feasible for certain models due to a list index out of range error.

Directions for Future Work:

- Applying hybrid normalization methods or exploring novel techniques such as GroupNorm.
- Employing lightweight architectures such as EfficientNet-Lite.
- Performing full fine-tuning rather than training only the head.
- Evaluation on more complex datasets such as Tiny-ImageNet.

7 Executive Summary

In this project, three different normalization structures were evaluated with and without gradient clipping in the context of transfer learning. The key findings are as follows:

- The FRN-NoClip structure achieved the highest accuracy (90.57%).
- Gradient clipping improved performance in the BatchNorm structure (+1.07%).
- LayerNorm and FRN generally delivered stable performance without the need for Clipping.

Responses to Three Key Questions:

1. **Best Normalization Technique:** FRN-NoClip, with a final accuracy of 90.57%.
2. **Effect of Clipping:** Positive and beneficial in BatchNorm; minimal or neutral in other methods.
3. **Practical Recommendations:** Employ LayerNorm/FRN in real-world projects with small batch sizes; enable Clipping when using BatchNorm; consider resource limitations when designing the model head.

8 References

References

Hao Li, Zheng Xu, Gavin Taylor, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, 2018.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

Aarush Singh and Dilip Krishnan. Filter response normalization layer: Eliminating batch dependence in the training of deep neural networks. In *CVPR*, 2020.

Zihan Zhang, Hao He, and Ruslan Salakhutdinov. Understanding and improving gradient clipping. *arXiv preprint arXiv:1905.11881*, 2019.