Compressing Deep Neural Networks with Knowledge Distillation: ResNet34 to ResNet18

Le Yu
University of Canterbury
Christchurch, New Zealand

***Abstract*—This project explores the application of knowledge distillation (KD) for compressing deep convolutional neural networks in image classification tasks. A pretrained ResNet34 is employed as the teacher model, and a lightweight ResNet18 is trained as the student model using an offline, response-based distillation strategy. The goal is to reduce inference cost while preserving classification performance. Experimental results show that the student model achieves 89% validation accuracy—a 9.23% drop compared to the teacher's 98.23%—while reducing model parameters by over 50% and improving inference speed by 2.4×. Detailed analyses of training dynamics, loss component analysis, and confusion matrices reveal that insufficient soft target entropy and model capacity constraints contribute to the performance gap. The study highlights the effectiveness and limitations of current KD settings, and proposes strategies such as temperature adjustment, loss reweighting, and architectural enhancements to improve distillation outcomes. This work provides practical insights into deploying efficient deep models on resource-constrained platforms.**

Keywords—Knowledge Distillation, Model Compression, ResNet, Image Classification, Soft Targets, Deep Learning Efficiency, Student-Teacher Framework

I. Introduction

In recent years, deep convolutional neural networks (CNNs) have achieved remarkable performance in tasks such as image classification. However, this high accuracy often comes at the cost of large network architectures and substantial computational resources, making it challenging to deploy such models on resource-constrained devices such as smartphones and embedded sensors. A key focus of current research is how to reduce model size and inference cost while maintaining strong classification performance.

To address this issue, Knowledge Distillation (KD) has gained significant attention as an effective technique for model compression and transfer learning. The core idea is to leverage a high-performing teacher model to guide the learning of a lightweight student model. Rather than learning solely from hard labels, the student model also learns from the teacher's output "soft targets," which capture richer inter-class relationships and improve generalization.

Building on this concept, the present project develops a knowledge distillation system for image classification. Specifically, a pretrained ResNet34 is used as the teacher model, and a ResNet18 is constructed as the student model. The student is trained under the supervision of the teacher through a distillation process. The goal is to retain classification performance as much as possible while significantly reducing computational cost.

The remainder of this report covers related work, methodology, experimental results, discussion, and conclusion.

II. Literature Review

Knowledge Distillation has become a widely adopted model compression technique in deep learning, aiming to compress large, high-performing models into smaller and more efficient ones for deployment on resource-constrained devices. The seminal work by Hinton et al. [1] introduced the concept of "soft targets," referring to the probability distribution output from the teacher model's softmax layer, smoothed by a temperature parameter. This distribution not only conveys the confidence in the correct class but also contains relative probabilities among incorrect classes, known as "dark knowledge." allowing the student to train more effectively with less data and a higher learning rate.

Subsequent studies have further categorized the types of "knowledge" transferred in KD, emphasizing that the process involves more than transferring parameters—it maps the function from input vectors to output predictions. Based on the form and source of knowledge, KD is generally classified into three types: (1) Response-based methods, as in Hinton's original proposal [1]; (2) Feature-based methods, such as the FitNets approach proposed by Romero et al. [2]; and (3) Relation-based methods, including those using pairwise distances or Gram matrices [3].

In terms of training paradigms, KD has evolved into several schemes. Offline distillation is the most common, where the teacher is pretrained and fixed. Online distillation, such as Deep Mutual Learning [4], allows the teacher and student to be trained jointly. Self-distillation refers to a network learning from its own earlier states or substructures, improving generalization [5].

To address the mismatch in capacity between large teachers and small students, Mirzadeh et al. proposed Teacher Assistant Knowledge Distillation (TAKD) [6], which introduces an intermediate-capacity assistant model to bridge the knowledge transfer. Experiments show that this significantly improves performance, especially for aggressively compressed students. In addition, Grathwohl et al. proposed the Joint Energy-based Model (JEM) [7], which reinterprets classifier logits as energy functions to jointly model the distribution of inputs and labels, enhancing both classification performance and robustness to out-of-distribution data.

Despite significant progress, several challenges remain in KD. For example, the theoretical understanding of why and how knowledge is effectively transferred—especially regarding improved generalization, remains incomplete. In cases of extreme capacity gap, distillation often fails unless aided by methods like TAKD [6], and even then, selecting the optimal assistant model structure remains an open problem. Moreover, the robustness and adaptability of

existing methods in cross-domain transfer and extreme compression scenarios are still limited.
In summary, the effectiveness of KD for deep model compression has been well-established, with diverse strategies and architectures proposed. In this study, I adopt an offline, response-based distillation strategy, using ResNet34 as the teacher model and ResNet18 as the student, aiming to explore the possibility of accelerating the model while maintaining acceptable accuracy.

III. Methodology

This section provides a detailed description of the technical approach used in this image classification and knowledge distillation task. It includes the overall distillation framework, data preprocessing pipeline, the construction and training of teacher and student models, the design of the distillation mechanism, and the selection and justification of key hyperparameters.

A. Knowledge Distillation Framework

The knowledge distillation framework adopted in this study follows an offline distillation strategy, consisting of three main components: the teacher model, soft targets, and the student model. The following describes the functionality and implementation of each module.

Teacher Model and Soft Targets

The *teacher model* is typically a large-capacity deep neural network that has achieved strong performance on the target task. In knowledge distillation, its primary role is to generate soft targets that serve as the learning signals for the student model. During distillation, the teacher's parameters remain fixed and do not participate in training.

Soft targets refer to the class probability distribution produced by the teacher model, i.e., the logits passed through a softmax function. Compared to one-hot encoded hard labels, soft targets provide relative confidence scores across classes, allowing the student model to learn inter-class similarities and structural relationships.

To control the smoothness of the softmax output, a temperature parameter τ is introduced. When τ >1, the output distribution becomes softer, which helps stabilize training and transfer richer information. Conversely, as $\tau \to 0$, the distribution approaches a hard, one-hot encoding.

The softmax output of the teacher model with temperature scaling is defined as shown in (1):

$$p_T^{(\tau)}(i) = \frac{\exp(z_i/\tau)}{\sum_{j=1}^{C}\exp(z_j/\tau)} \quad (1)$$

where z_i denotes the logit of class i, C is the total number of classes, and τ is the temperature parameter that controls the softness of the output distribution.

Student Model

The *student model* is typically a compact neural network with fewer parameters and lower computational cost, making it well-suited for scenarios with limited resources or strict latency requirements. Within the distillation framework, the student is trained using both the soft targets generated by the teacher and the original hard labels, aiming to approximate the teacher's predictive behavior without increasing model complexity.

Distillation Loss Function

The distillation loss serves as the central training objective in the knowledge distillation framework. It enables the student model to learn from both the teacher's soft targets and the ground-truth hard labels. This loss function typically consists of two components:

- **Kullback–Leibler (KL) Divergence Loss**: This component measures the divergence between the student's softmax output and the teacher's output after temperature scaling. It encourages the student to mimic the teacher's predictive distribution as shown in (2).

$$\text{KL loss} = \text{KLDivLoss}\left(\text{softmax}\left(\frac{s}{T}\right), \text{softmax}\left(\frac{t}{T}\right)\right) \times T^2 \quad (2)$$

To compensate for the gradient scaling effect introduced by temperature smoothing, I follow the formulation proposed by Hinton et al.[1], where the KL divergence term is multiplied by T^2 . This adjustment is designed to counteract the dilution of information caused by the softened probabilities.

- **Cross-Entropy Loss (Hard Label Loss):** This term measures the discrepancy between the student model's predictions and the ground-truth labels. It ensures that, while learning to mimic the teacher, the student does not deviate from the true data distribution, thereby preserving its ability to model the core classification task.

The two loss components are combined using a weighting factor $\alpha \in [0,1]$, resulting in the final joint training objective as shown in (3):

$$\mathcal{L}_{KD} = \alpha \cdot \text{KL}\left(p_T^{(\tau)} \parallel p_S^{(\tau)}\right) + (1-\alpha)\cdot \text{CE}(y, p_S) \quad (3)$$

Where $p_T^{(\tau)}$ and $p_S^{(\tau)}$ denote the temperature-scaled softmax outputs of the teacher and student models, respectively, and y represents the ground-truth label.

B. Teacher and Student Architectures

In this study, the ResNet (Residual Network) family is employed for both the teacher and student models in the knowledge distillation task. Originally proposed by He et al. [8], ResNet introduces residual connections to mitigate the vanishing gradient problem in deep networks, thereby enabling the effective training of very deep architectures.

- **ResNet34 (Teacher Model):** This model consists of 34 layers and stacks multiple residual blocks, each comprising 3×3 convolutional layers and identity shortcut connections. It contains approximately 21.8 million parameters and has demonstrated strong generalization performance on large-scale image classification benchmarks such as ImageNet. In this experiment, the teacher model was initialized with pretrained weights from ImageNet to leverage the learned feature representations and improve training efficiency.
- **ResNet18 (Student Model)**: Designed for lightweight deployment, ResNet18 contains 18 layers and approximately 11.7 million parameters, roughly half the size of ResNet34. Despite its reduced complexity, it has consistently shown a strong trade-off between accuracy and efficiency in various model compression

tasks, making it well-suited for deployment in resource-constrained environments.

C. Data Preprocessing

To ensure consistency in model inputs and stability during training, all raw image data underwent standardized preprocessing. The dataset contains 10 classes, with 780 training and 260 validation images per class, and 2,600 images in the test set. The preprocessing steps are detailed as follows:

- **Image Resizing:** All images were resized to 224×224 resolution to match the input size requirements of ResNet models. This dimension is consistent with the standard input format for ImageNet-pretrained models and facilitates transfer learning. Images smaller than 224 pixels on any side were padded symmetrically, while larger images were center-cropped to fit the target size.
- **Normalization and Standardization**: Image pixels were first converted to float32 and normalized to the range [0, 1] to improve numerical stability and accelerate convergence. Then, each RGB channel was standardized independently by subtracting the mean and dividing by the standard deviation, so that the channel-wise distribution had a mean of 0 and standard deviation of 1.
- **Channel Order Conversion**: To comply with PyTorch's tensor structure, the image dimension order was converted from NHWC (Height, Width, Channel) to NCHW (Channel, Height, Width). All images were then transformed into tensor format as model input.
- **Saving Preprocessed Data:** All preprocessed data were saved in .pt tensor format to enable efficient reuse and reproducibility during subsequent training and evaluation stages.

D. Teacher Model Training Strategy and Hyperparameter Configuration

In this study, ResNet34 was used as the teacher model, initialized with pretrained weights from the ImageNet dataset. To adapt the model to the current 10-class classification task, the final fully connected layer was replaced with a new output layer matching the number of classes. The rest of the model architecture remained unchanged, and fine-tuning was performed on the task-specific dataset.

As the model achieved a 98% accuracy on the validation set using the initial configuration, no extensive hyperparameter tuning was conducted. The training strategy is summarized as follows:

- **Optimizer and Regularization:** The model was trained using the Adam optimizer (with momentum = 0.9) and a weight decay of 1×10^{-4}. This configuration is commonly used in distillation literature (e.g., TAKD [6]) and has been shown to work well in fine-tuning and small-sample classification tasks.
- **Learning Rate Scheduling**: The initial learning rate was set to 0.001, with a StepLR scheduler that decays the learning rate by a factor of 0.1 every 10 epochs. This setup led to fast convergence in our task. While Hinton et al. [1] introduced the temperature mechanism for distillation, their paper does not specify concrete hyperparameter values. My choices were guided by empirical performance.
- **Batch Size and Number of Epochs**: A batch size of 128 was selected as an efficient setting within available resource constraints. This value is also frequently used in distillation experiments. The model was trained for a maximum of 20 epochs, with early stopping (patience = 5) to prevent overfitting.

A summary of the training configuration is provided in Table I.

Table I. Training Configuration for the Teacher Model

Parameter	Value
Learning Rate (LR)	0.001
Optimizer	Adam (momentum = 0.9)
Weight Decay	1e-4
Step Size	10
gamma	0.1
Batch Size	128
Epochs	20
Early Stopping	Patience = 5

E. Student Model Training Strategy and Hyperparameter Configuration

An untrained ResNet18 was used as the student model in this study. During training, the parameters of the teacher model were kept frozen, and only the student model's weights were updated. The training procedure included the following key components:

- **Optimizer and Regularization**: The student model was trained using the Adam optimizer with a learning rate of 0.001 and weight decay of 1×10^{-4}. This configuration has demonstrated strong convergence performance in lightweight models trained from scratch and is consistent with settings adopted in distillation literature such as TAKD [6].
- **Learning Rate Scheduling and Training Epochs**: A StepLR scheduler was applied to reduce the learning rate by a factor of 0.5 every 10 epochs. This gradually lowers the learning rate during mid-to-late training, improving stability and convergence. The total number of training epochs was set to 80, as the student model was trained from scratch (without pretrained weights) and required a longer training period to fully converge.
- **Batch Size and Early Stopping Strategy**: A batch size of 64 was used—smaller than that of the teacher model—to accommodate the computational limits and memory constraints of CPU-based training. Early stopping was enabled and paired with a dual-metric monitoring mechanism, using validation accuracy as the primary metric and validation loss as a secondary criterion. Training was terminated early if both metrics showed no improvement for 10 consecutive epochs, to avoid overfitting and reduce unnecessary computation.
- **Distillation Loss Function**: The student model was trained using a weighted combination of cross-entropy loss and KL divergence (see Distillation Loss Function section). The soft targets were obtained from the teacher model's output and smoothed using a

temperature T=4. The total loss was computed as in Equation (3):

$$\mathcal{L}_{\mathrm{KD}} = \alpha \cdot \mathrm{KL}\left(p_T^{(\tau)} \parallel p_S^{(\tau)}\right) + (1-\alpha) \cdot \mathrm{CE}(y, p_S)$$

The distillation weight α was set to 0.7 to emphasize the teacher's supervision signal. This value has been validated in multiple classic distillation studies.

- **Hardware Constraints:** All experiments were conducted **without GPU acceleration**, using only CPU resources. As such, model architecture choices, batch size, and training strategies were carefully adjusted to balance computational efficiency with performance.

Table II. Training Configuration for the Student Model

Parameter	Value
Epochs	80
Optimizer	Adam
Learning Rate	0.001
Batch Size	64
Early Stopping	patience = 10
Temperature (c)	4
Distillation Weight (α)	0.7

IV. Results

This section presents the outcomes of the knowledge distillation task, evaluating the classification performance, model efficiency, and the effectiveness of the student model compared to the teacher model..

A. Classification Accuracy and Evaluation Metrics

Overall Accuracy

Under the current distillation configuration, the student model achieved a classification accuracy of 89.81% on the validation set (see Table III), whereas the teacher model reached an accuracy of 98.23%.

Table III. Comparison of Teacher and Student Model Performance

Model	Accuracy (%)	Total Loss
Teacher	98.23	0.0610
Student	89.81	0.7094

Per-Class Accuracy

To further investigate the source of performance degradation in the student model, I compared its classification accuracy across individual classes.

As shown in Fig.1, the teacher model maintained consistently high accuracy across all classes (most above **97%**), whereas the student model exhibited noticeable drops in categories such as *sunglasses* and *parachute*.

This suggests that knowledge distillation may provide insufficient supervisory signals for classes with complex visual patterns or high inter-class similarity, leading to reduced generalization performance.

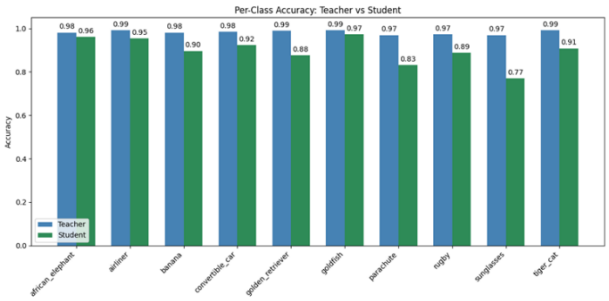

Fig. 1. Per-class accuracy comparison between teacher and student models

Confusion Matrix Analysis

The confusion matrix in Fig. 2 shows that the darkest regions are concentrated along the diagonal, indicating that the student model achieves high classification accuracy across most categories. For each class, more than 200 samples are correctly classified, reflecting the model's overall recognition capability.

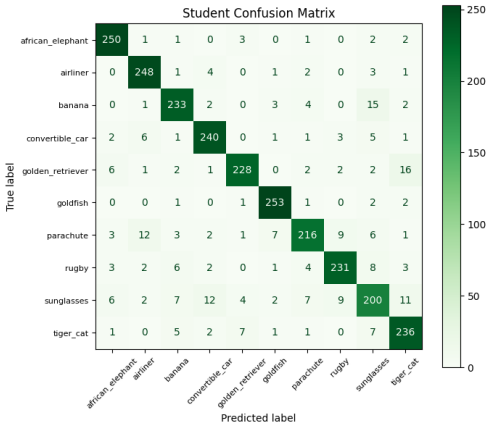

Fig. 2. Confusion matrices for the student model.

However, several notable misclassification patterns were observed:

Sunglasses was the most misclassified category, with a total of 60 incorrect predictions. Among these, 12 were misclassified as *convertible_car* and 11 as *tiger_cat*. This suggests that *sunglasses* may share visual similarities—either in shape or background—with these two categories, leading to confusion.

In the *parachute* class, 44 samples were misclassified, with 12 incorrectly labeled as airliner, making it one of the most frequent misclassification directions. Both *parachute* and *airliner* images often contain the sky as a background, which may confuse the model due to similar contextual features.

The *banana* class had 27 misclassifications, 15 of which were incorrectly labeled as sunglasses. This may be attributed to visual similarities between the two classes in some images—for instance, the curved shape of a banana may resemble the contour of *sunglasses* from certain angles.

These specific misclassification patterns help identify the student model's "weak classes" and provide valuable insight for refining future distillation strategies—such as class reweighting or targeted data augmentation.

B. Training Dynamics and Loss Breakdown

Accuracy Curves over Epochs

To provide a clearer picture of training behavior and model convergence, Fig. 3 presents the accuracy curves on both the training and validation sets.

The training accuracy increases rapidly within the first 10 epochs and approaches saturation around epoch 25, stabilizing near 100%, indicating that the student model has closely fit the training data.

In contrast, the validation accuracy exhibits noticeable fluctuations between epochs 10 and 20, suggesting that the model is still adjusting its generalization capability—potentially influenced by the soft target guidance and the limited model capacity. After epoch 25, validation accuracy plateaus at around 89%, failing to improve in parallel with training accuracy.

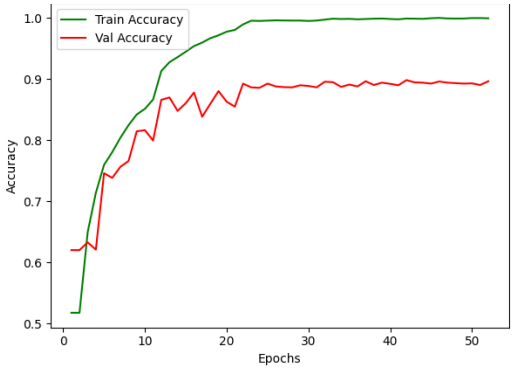

Fig. 3. Accuracy across training epochs for the student model.

The final gap of approximately **9%** between validation and training accuracy indicates the presence of generalization error, highlighting room for further optimization.

Loss Curves (Total / CE / KD)

As shown in Fig. 4, the total loss on both training and validation sets drops rapidly within the first 20 epochs and then stabilizes, indicating that the student model is progressively fitting the training data while learning from the teacher's soft targets. The **training loss** continues to decrease to a very low level, while the **validation loss** plateaus after epoch 35 without further decline. This aligns with the saturation observed in the validation accuracy curve, suggesting a tendency toward overfitting.

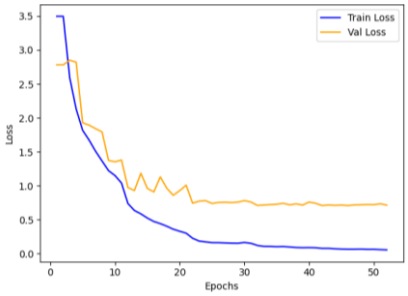

Fig. 4. Loss curve across training epochs for the student model.

Training and Validation Loss Breakdown

The breakdown of KD Loss and CE Loss provides further insight into the student model's learning dynamics:

- **Training Set**: As shown in Fig. 5, the KD loss starts significantly higher than the CE loss but decreases more rapidly, approaching the CE loss around epoch 20. This suggests that the student is not only fitting the ground-truth labels well but also effectively learning from the teacher's soft labels. The convergence of both losses in later epochs indicates stable distillation and improved alignment with the teacher's output distribution.

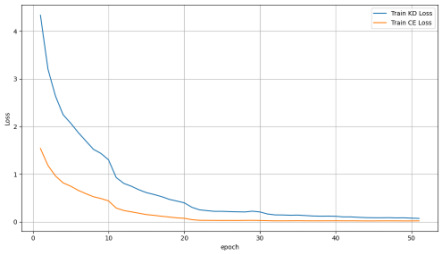

Fig. 5. Training KD loss and CE loss over epochs.

- **Validation Set**: As shown in Fig. 6, the CE loss decreases steadily in early training and stabilizes at a lower level than KD loss, suggesting better generalization to ground-truth labels. In contrast, the KD loss stabilizes at a relatively high level after epoch 20 and does not align with the training loss. This implies that the student struggles to generalize the teacher's soft targets, pointing to potential room for improvement in temperature T and loss weight α settings.

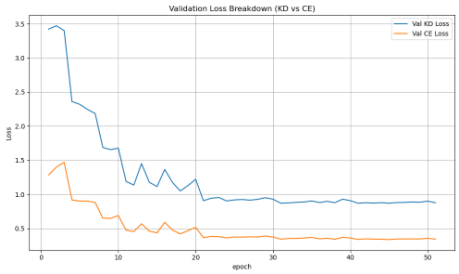

Fig. 6. Validation KD loss and CE loss over epochs.

C. Soft Target Analysis（T=4）

Fig. 7 shows that the teacher's maximum softmax probabilities mostly fall between 0.4 and 0.5, indicating that the outputs are effectively softened. Fig. 8 reveals an entropy peak around 1.8–1.9, which is close to the theoretical maximum of $log(10) \approx 2.3$, suggesting that the teacher's predictions retain sufficient uncertainty.

These results confirm that the temperature setting T=4 yields informative soft targets.

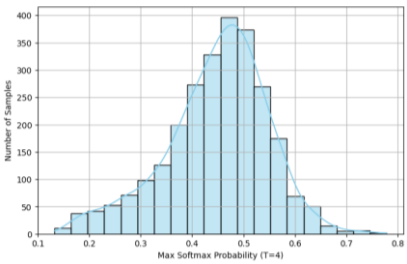

Fig. 7. Teacher Model Confidence Distribution(T=4)

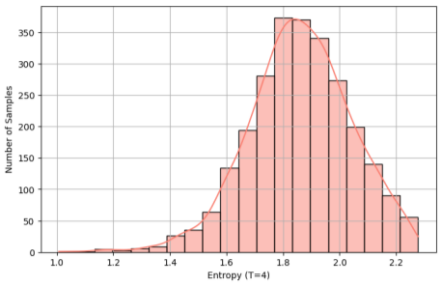

Fig. 8. Teacher Model Output Entropy Distribution(T=4)

D. Computational Efficiency

The student model demonstrates a clear efficiency advantage over the teacher, with approximately half the number of parameters and model size, and a moderately faster inference speed (75.97 ms vs. 95.77 ms per image), as shown in Table IV.

Table IV. Model Efficiency Comparison Between Teacher and Student

Model	Params (M)	Model Size (MB)	Inference Time (ms/img)
Teacher ResNet34	21.8	83.3	95.77
Student ResNet18	11.7	44.7	75.97

V. DISCUSSION

In this experiment, a pretrained ResNet34 was distilled into a smaller ResNet18 model to improve inference efficiency while preserving classification performance. The teacher model achieved 98.23% validation accuracy, while the student reached only 89%, revealing a performance gap of over 9 percentage points. This indicates that the current distillation strategy did not fully transfer the teacher's knowledge, and the student model faces generalization challenges.

Training dynamics show that the student quickly fit the hard labels, with cross-entropy loss converging early. However, the distillation loss remained high, suggesting insufficient absorption of the teacher's soft labels.

The confusion matrix and per-class accuracy analyses revealed significant misclassifications in confusing pairs like sunglasses vs. banana and parachute vs. airliner. These errors highlight the student model's limited ability to capture fine-grained semantic differences, especially in visually similar categories. This limitation may stem from both the quality of soft targets and the student model's representational capacity.

These findings suggest the following directions for improvement:

- Raise the KD loss weight α to strengthen the role of teacher guidance;
- Adopt more adaptive LR schedulers to respond to subtle convergence changes;
- Enhance model architecture with lightweight modules like attention or skip connections;
- Apply error-aware loss weighting to focus training on difficult class boundaries.

The current design also has limitations. Due to CPU-only constraints, the model was trained once with theoretically suggested parameters, without further tuning. With GPU acceleration, additional fine-tuning could likely enhance the results.

Hyperparameters such as T and α were selected empirically rather than tuned systematically. Future work could employ grid search or Bayesian optimization. The training strategy used single-stage offline distillation, and more advanced schemes like progressive or self-distillation could be explored. Additionally, data quality was not deeply assessed; future studies may incorporate augmentation or filtering to mitigate class imbalance or noise.

Overall, this experiment deepened my understanding of the challenges of knowledge distillation. Effective knowledge transfer depends not just on loss formulation, but on the interaction between soft label quality, model capacity, and training strategy. These insights will inform future work on model compression and efficient deep learning deployment.

VI. CONCLUSION

This study implemented knowledge distillation by compressing a pretrained ResNet34 into a lightweight ResNet18 model and evaluated their performance on an image classification task. While the student model achieved 89% validation accuracy—falling short of the teacher's 98.23%—it preserved strong recognition ability across most classes, demonstrating the initial effectiveness of the distillation approach in reducing model complexity.

Training curves showed that the student rapidly fit the hard labels, while its alignment with the teacher's soft labels remained suboptimal, as indicated by persistently high KD loss. Analysis of soft target distributions and confusion patterns revealed that low entropy and ambiguous class boundaries were key limitations to generalization.

To address these issues, future work should consider adjusting the loss weight allocation to emphasize KD supervision, and improving the student architecture with lightweight enhancements such as attention or skip connections. Additionally, class-specific reweighting may help mitigate misclassifications in semantically similar categories.

In conclusion, this project confirms the potential of knowledge distillation for efficient model compression, while also highlighting the importance of well-designed soft targets, adaptive training strategies, and architectural capacity in achieving effective knowledge transfer.

REFERENCES

[1] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv preprint arXiv:1503.02531, 2015.

[2] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," in *Proc. ICLR*, 2015.

[3] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proc. CVPR*, 2017, pp. 4133–4141.

[4] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, "Deep mutual learning," in *Proc. CVPR*, 2018, pp. 4320–4328.

[5] X. Zhang, Z. Li, C. Liu, and D. Lin, "Self-distillation as instance-specific label smoothing," arXiv preprint arXiv:1909.09148, 2019.

[6] S. Mirzadeh, M. Farajtabar, A. Li, and H. Ghasemzadeh, “Improved knowledge distillation via teacher assistant,” in *Proc. AAAI*, 2020, pp. 5191–5198.

[7] W. Grathwohl, K. K. Srinivasan, D. Duvenaud, J. Schwarz, and Y. W. Teh, “Your classifier is secretly an energy based model and you should treat it like one,” in *Proc. ICLR*, 2020.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.