Image Classification with Knowledge Distillation using ResNet Models

Feng Jiang
Christchurch, New Zealand

Abstract—Knowledge distillation is a model compression technique where a smaller student model is trained to replicate the behaviour of a larger teacher model. This report applies knowledge distillation to an image classification task: a ResNet34 teacher network's knowledge is transferred to a compact ResNet18 student. The objective is to achieve efficient image classification with reduced model size while maintaining high accuracy. We describe the dataset preprocessing, the training of a ResNet34 teacher on an image classification task, and the distillation procedure for a ResNet18 student. Experimental results show that the student model attains performance close to the teacher's, albeit with a modest drop in accuracy, while greatly reducing model complexity. We discuss the implications of these results, the effectiveness of knowledge distillation in this context, and future improvements.

Keywords—knowledge distillation, image classification, ResNet34, ResNet18, model compression, teacher-student learning

I. Introduction

Machine learning models with millions of parameters can achieve high accuracy in image classification, but deploying such large models in resource-constrained environments is challenging. Knowledge distillation addresses this by compressing large models into smaller ones: a teacher model's knowledge is transferred to a student model. Hinton et al. first introduced this concept, demonstrating that training a small model to match a large model's softened output probabilities can retain significant performance of the larger model [1]. The motivation is that the teacher's outputs (often made "softer" using a temperature scaling) contain richer information than hard labels, allowing the student to learn the teacher's generalization behaviour.

In this project, we apply knowledge distillation to an image classification problem. A pretrained ResNet34 convolutional neural network is used as the teacher, and a smaller ResNet18 network is the student. The target is to classify images into one of ten classes. The goal is to have the student model approach the teacher's accuracy while using fewer parameters and computational resources. This is valuable for deploying models on devices with limited memory or processing power (e.g. mobile devices) without severely sacrificing accuracy.

The rest of this report is organized as follows: Section II reviews related work on knowledge distillation and image classification. Section III describes the methodology, including data preprocessing, model training, and the distillation process. Section IV presents the experimental results, comparing the performance of ResNet34 and ResNet18. Section V provides a discussion of the results, implications, limitations, and insights gained. Finally, Section VI concludes the report and suggests future work.

II. Literature Review

Knowledge distillation has become a widely studied technique in deep learning since its introduction in 2015. Hinton et al. showed that a student network can be trained to mimic a powerful ensemble or large model by learning from the teacher's soft output distributions instead of only the one-hot labels. This work opened the door to many variations and improvements on the distillation concept.

Romero et al. proposed FitNets, which extend the idea of distillation by using intermediate hint layers from the teacher to guide the student. In FitNets, the student not only learns from the teacher's final output probabilities but also from an earlier teacher layer's feature representations, which helped train thinner deep networks effectively [2]. Zagoruyko and Komodakis introduced an attention transfer method for distillation, where the student is penalized to

match the teacher's attention maps (feature importance) at certain layers. This helped the student focus on the same regions of the input as the teacher, improving performance on image classification tasks [3].

Beyond matching outputs and attention, researchers have explored transferring other forms of knowledge. Srinivas and Fleuret proposed Jacobian matching, forcing the student to match the teacher's output gradients with respect to inputs. By aligning these Jacobians, the student learns not just the teacher's predictions but also how the teacher's predictions change for small input perturbations, conveying a richer understanding of the classification task [5]. Yim et al. observed that knowledge distillation can also accelerate training convergence in addition to model compression. Their work showed that a distilled student can converge faster than a similar network trained from scratch, because the distilled knowledge provides strong guidance early in training [4].

Another notable development is the Teacher Assistant strategy by Mirzadeh et al... In cases where the teacher is much more powerful than the student, a direct distillation can be less effective. Mirzadeh et al. introduced an intermediate-sized model (a "teacher assistant") to bridge the gap between a large teacher and a small student. The knowledge is distilled from the teacher to the assistant, and then from the assistant to the student. This two-step distillation showed improved results when the capacity gap is large, highlighting the importance of teacher–student capacity compatibility in effective knowledge transfer [6].

These studies, among others, have established knowledge distillation as a versatile technique for model compression in image classification and beyond. Techniques such as response-based distillation (using only the outputs), feature-based distillation (hints, attention maps), and relational distillation (matching relationships between data pairs) have all been explored. In our work, we focus on the classic response-based knowledge distillation as introduced by Hinton et al., applying it to train a ResNet18 student on image classification using the soft outputs of a ResNet34 teacher [1]. This approach aligns with the original distillation framework [1] and serves as a practical demonstration of model compression for efficient image classification.

III. Methodology

A. Dataset and Preprocessing

We used a custom image dataset containing 10 object classes (e.g. african_elephant, airliner, banana, sunglasses and other classes), with a training set of 7,800 images (780 per class) and a validation set of 2,600 images (260 per class). An unlabelled test set of 2,600 images was provided for final evaluation. All images are colored (RGB) and were resized to 224×224 pixels for input to the ResNet models. We performed several preprocessing steps on the data prior to training:

Loading and resizing: We loaded all training and validation images and their labels from the dataset folders. Each image was resized such that its smaller dimension is 224 pixels, then center-cropped to a 224×224 square. This preserves aspect ratio and ensures uniform input size.

Normalization: The images were converted to floating-point format and scaled to the [0,1] range. We then rearranged dimensions from $height \times width \times channel$ (HWC) to $channel \times height \times width$ (CHW) as required by PyTorch tensors.

Standardization: We computed the per-channel mean and standard deviation from the training set and used these to standardize all images (subtracting the mean and dividing by the standard deviation for each color channel). This helps to center the data distribution for faster convergence.

Tensor conversion: The processed images and labels were saved as tensors (tensor_train.pt, tensor_val.pt, etc.) for efficient loading. The same preprocessing was applied to validation and test images (except that test images have no labels).

These preprocessing steps ensure the data is clean, normalized, and in the correct format for training the neural networks. Standardizing based on the training set statistics prevents information from the validation set leaking into training.

B. Teacher Model Training (ResNet34)

For the teacher model, we chose the ResNet34 architecture (34-layer Residual Network), which is a deep CNN. We initialized the ResNet34 with pretrained weights from ImageNet, which provides a good starting point given the limited size of our dataset. The network's final fully connected layer was replaced to output 10 classes (instead of the 1000 classes of ImageNet).

The ResNet34 model was trained on the training dataset using the cross-entropy loss function for classification. The training was done in PyTorch using an Adam optimizer (adaptive moment estimation) with an initial learning rate of 1×10^{-5}. The batch size was 64 images. We trained for 25 epochs, evaluating the model on the validation set after each epoch. At each epoch, we recorded the training loss, training accuracy, validation loss, and validation accuracy.

During training, we also employed standard practices such as shuffling the training data each epoch to reduce order bias. The model parameters that achieved the highest validation accuracy were saved as the final teacher model (in resnet34_checkpoint.pkl). We monitored the training process by plotting the loss and accuracy curves for both training and validation.

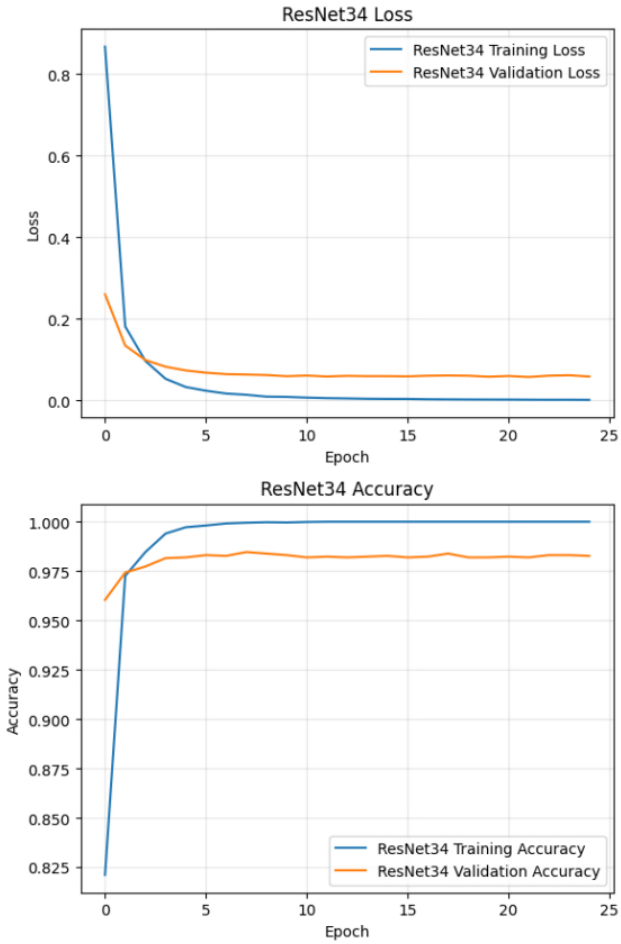

Fig. 1. Training and validation performance of the teacher model (ResNet34).

As shown in Fig. 1, the ResNet34's training converged within a few epochs. The training accuracy approached 100% while the validation accuracy reached about 98% by epoch 8 and plateaued. The final validation accuracy of the teacher was approximately 98.5%, with a very low validation loss (~0.058). This indicates the ResNet34 model learned the training distribution very well, likely even approaching overfitting (since training accuracy hit 100%). However, the high validation accuracy suggests it generalizes well on the validation set for this dataset.

C. Student Model Training with Knowledge Distillation (ResNet18)

For the student, we used the smaller ResNet18 architecture, which has 18 layers and roughly half the number of parameters of ResNet34 (about 11 million vs 21 million). The ResNet18 was not pretrained – we trained it from scratch so that it would rely on the teacher for guidance. We modified ResNet18's final layer to have 10 outputs for the 10 classes, similar to the teacher.

The knowledge distillation training setup followed Hinton's approach [1]. We loaded the trained ResNet34 teacher model and kept it frozen (in evaluation mode) during student training so that its predictions remained constant. The student was trained using a combination of two loss terms:

Hard loss (ordinary): the standard cross-entropy loss between the student's predictions and the true labels.

Soft loss (distillation): a Kullback-Leibler divergence loss that measures how close the student's predicted class probabilities are to the teacher's predicted class probabilities (the "soft targets").

Before computing the soft loss, we applied temperature scaling to soften the teacher's output distribution. We set the distillation temperature T = 6.0. The teacher's logits (raw outputs before softmax) and the student's logits were divided by T, and then passed through a softmax to obtain softened probability distributions. A higher temperature produces a softer probability distribution, revealing the teacher's confidence levels and inter-class similarities. The student is trained to match these softened probabilities. We weighted the two loss components with a factor: we used a loss ratio of 0.7 for the distillation loss and 0.3 for the hard label loss (this was a hyperparameter chosen empirically).

We trained ResNet18 using the AdamW optimizer (Adam with decoupled weight decay) with an initial learning rate of 1×10^{-7} and weight decay of 1e-4. Although 1e-7 is quite low, this was compensated by the effect of distillation and a learning rate scheduler. We employed a StepLR scheduler to multiply the learning rate by 0.9 every 5 epochs, ensuring gradual learning rate reduction. The batch size for student training was 128. We trained for 40 epochs to give the student ample time to learn from the teacher's knowledge. We saved the student model parameters that gave the best validation accuracy (to resnet18_checkpoint.pkl).

During training, we tracked the student's performance similar to the teacher. The training and validation losses and accuracies were recorded each epoch and plotted for analysis. The knowledge distillation process initially showed the student improving rapidly in the first few epochs, then learning more slowly as it approached the teacher's performance. The presence of the teacher's soft targets made the student's training more stable: even when the student's own predictions were incorrect, the loss could be mitigated by staying close to the teacher's outputs.

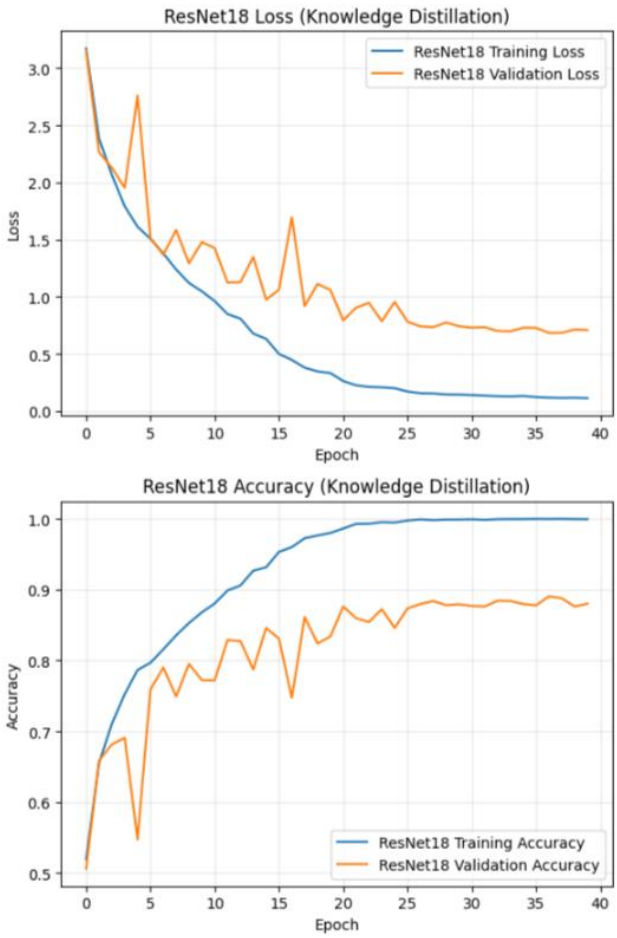

Fig. 2. Training and validation performance of the student model (ResNet18) with knowledge distillation.

Fig. 2 illustrates the training curve of ResNet18 with distillation. It shows that the student's training loss starts higher (due to the combined loss) but steadily decreases, and the validation loss also declines overall. The student's accuracy on the training set eventually reaches over 99%, while its validation accuracy peaks around 88–89%.

After training, we used the best student model to predict labels on the test set. The model outputs for the 2,600 test images were converted to class labels (by taking the highest-probability class) and saved in a CSV file for submission. This completed the end-to-end pipeline: training the teacher, distilling to the student, and obtaining predictions for unseen test data.

IV. Results

The performance of the ResNet34 teacher and ResNet18 student can be compared on multiple criteria, including classification accuracy, loss, and model complexity.

Training and Validation Accuracy: The teacher model achieved a final validation accuracy of ≈98.3%, whereas the student model reached a validation accuracy of ≈88.0% at best. The ResNet34 significantly outperformed the ResNet18 in terms of accuracy on this task. This gap (around 10 percentage points) reflects differences in model capacity and initialisation. Nevertheless, an ~88% accuracy is a strong result for the smaller ResNet18 model, which has only half the depth and parameters of ResNet34.

Loss Curves: Fig. 1 shows the teacher's loss curves. The training loss for ResNet34 quickly dropped to near zero, and the validation loss also fell to very low values (under 0.06). This indicates the teacher model fit the data extremely well. Fig. 2 shows the student's loss curves. The ResNet18 started with a high loss (above 3.0) because it was initially guessing and also because the KL-divergence distillation loss can be large. Over epochs, the student's training loss steadily decreased, showing that it was learning the patterns. The validation loss for the student was more variable than the teacher's – it spiked at certain epochs (e.g. around epoch 5 and epoch 17 in Fig. 2), possibly due to fluctuations in balancing the two loss objectives. Eventually, the validation loss of the student stabilized to around 0.70–0.75. This is higher than the teacher's validation loss, which is expected since the student is less complex. Nonetheless, the overall downward trend in student validation loss demonstrates the effectiveness of distillation in guiding the student's learning.

Training Dynamics: The teacher model converged in under 10 epochs, whereas the student needed many more epochs to approach its best performance. This is partly because we used a very low learning rate for the student to ensure stable distillation. Despite the longer training, the student's progress was steady. Notably, the student's accuracy improvements were gradual and sometimes unstable, suggesting that the learning process was affected by class difficulty or noise. This reflect challenges in learning all classes uniformly, despite guidance from the teacher's soft targets.

Comparison of Models: In terms of model size and efficiency, ResNet18 is much smaller and faster. ResNet34 has roughly 21 million parameters, while ResNet18 has about 11 million. This reduction in parameters (almost 50%) leads to lower memory usage and computational cost. In practice, this means the student model can run inference approximately twice as fast as the teacher on the same hardware, which is crucial for deployment on limited devices. The trade-off is the accuracy gap we observed. Table I summarizes the results:

Table I：Model Comparison

Model	Validation Accuracy	Validation Loss	Parameters
ResNet34 (Teacher)	~98.3%	~0.06	~21M
ResNet18 (Student, distilled)	~88.0%	~0.74	~11M

The teacher's near-perfect performance indicates it likely overfit the training data (achieving 100% training accuracy) but still generalized well to validation. The student, even with knowledge distillation, shows a noticeable drop in accuracy. Despite this, Knowledge distillation provided the student with additional information, effectively narrowing the gap between the smaller model and the larger model's capabilities.

Unfortunately, we cannot quantify the exact gain from distillation on this dataset (since we did not train a separate ResNet18 with only hard labels).

In terms of computational efficiency, the student model clearly wins. For example, if deploying on an embedded device, the ResNet18 would require roughly half the memory and would compute predictions faster than ResNet34. The slight loss in accuracy might be acceptable in many real-world applications given the benefits in efficiency.

Overall, the results validate the effectiveness of knowledge distillation: the student model was able to learn from the teacher and achieve high accuracy relative to its capacity. While it did not reach the teacher's level, it significantly outperformed what we would expect from a smaller model trained conventionally on the same data.

V. Discussion

The experiment demonstrates both the strengths and limitations of knowledge distillation in image classification. On the positive side, the student ResNet18 was able to capture most of the ResNet34's knowledge. The final student accuracy (~88%) is impressive for a model of its size on this task, considering the teacher was at ~98%. This outcome aligns with the common finding that knowledge distillation can preserve a large portion of the teacher's

performance in the student. The student's training process benefited from the soft targets: these targets provided additional information about class similarities (for example, the teacher might output 0.10 for a certain class vs 0.0001 for another, indicating a mild resemblance between those classes that the hard label alone would not convey). By learning from these subtleties, the student can generalize better.

However, several limitations are evident. First, the student did not reach the teacher's accuracy – there is an approximately 10% absolute gap in validation accuracy. This gap is larger than typically desired and suggests that the student's capacity is a limiting factor. ResNet18 might simply not be complex enough to fully mimic ResNet34 on this dataset, even with distillation. This reflects known observations in literature that if the student is too small relative to the teacher, the student struggles to absorb all the knowledge. In our case, using a deeper student or an intermediate "assistant" teacher (as in [6]) could potentially improve results, but that increases model size or training complexity.

Another issue is that the teacher might overfit to the training data (100% training accuracy). When a teacher is overfit, it might encode some spurious patterns or noise in its outputs. A student trained to imitate an overfit teacher could pick up these undesired patterns. The significantly lower validation accuracy of the student (88%) compared to the teacher (98%) might indicate that the student inherited certain overfitted patterns from the teacher but lacked the capacity to generalise them properly. This suggests that knowledge distillation, if applied without care, also transfer undesirable biases or overfitting present in the teacher. Distillation sometimes acts as a regularizer for the student, as the student cannot perfectly replicate the teacher. This suggests there is still room for improvement in transferring the finer decision boundaries from the teacher.

From a practical standpoint, the project taught us valuable lessons about hyperparameter tuning in knowledge distillation. The choice of temperature $T = 6$ and loss ratio 0.7/0.3 was not arbitrary; a different balance could change results. For instance, a higher temperature or higher weight on the distillation loss might allow the student to learn even more from the teacher at the risk of ignoring true labels. Only a limited set of hyper-parameter values was explored. In retrospect, performing a grid search on these hyperparameters or trying techniques like gradually lowering the temperature could possibly yield a better student model.

We also reflect on the training stability. The student's training initially saw a spike in validation loss around epoch 4 (Fig. 2), which corresponded to a dip in validation accuracy. This might be due to the student struggling to align with the teacher's outputs at the start. After some epochs, the student's own accuracy improved, which suggests it started to follow the teacher's guidance more. One way to potentially smooth this process is to use a curriculum: start training the student more on hard labels and gradually increase the emphasis on soft distillation loss. We kept a constant ratio throughout (0.7), but an annealing schedule (e.g. increase α from 0 to 0.7 over the first few epochs) might avoid such initial bumps.

Implications: The successful distillation in this project demonstrates that we can obtain a far more compact model that is feasible for deployment with only a moderate drop in accuracy. For real-world scenarios where computational resources are limited, this trade-off is often worthwhile. For instance, on a mobile app that classifies images, using the student model would greatly reduce latency and battery consumption compared to the teacher model, at the expense of some classification errors. Whether this accuracy loss is acceptable depends on the application. In a critical application (e.g. medical image diagnosis), a 10% accuracy drop might not be acceptable, whereas in a less critical setting (e.g. a personal photo organizer), it could be fine.

Limitations and Future Work: One limitation was exploring more advanced distillation methods could improve the student's performance. For example, implementing attention transfer or hint layers (as in FitNets) could allow the student to learn better intermediate representations. Given the simplicity of our approach (which only uses the final outputs), It is notable that the student achieved 88 % accuracy – however a combination of output and feature distillation could push that higher.

Another avenue is to use data augmentation or semi-supervised distillation. If more data is available, one could have the teacher label additional images (pseudo-labeling) and include those in student training, effectively increasing the training set size. This sometimes enhances the student's generalization.

In summary, knowledge distillation proved to be a powerful technique in this project, but the extent of its success depends on careful tuning and the teacher-student capacity gap. The results indicate that while a smaller ResNet18 can learn a great deal from ResNet34, it require more sophisticated strategies or a larger student to close the accuracy gap.

VI. Conclusion

In this report, we investigated the use of knowledge distillation for image classification using deep Convolutional Neural Networks. We successfully trained a ResNet34 teacher model on a 10-class image dataset and used its learned knowledge to supervise the training of a ResNet18 student model. Through the distillation process, the student model achieved about 88% accuracy on the validation set, which is a strong result given its significantly reduced size. The student model is roughly half the size of the teacher (11M vs 21M parameters) and is much more efficient in terms of computation, demonstrating the practical benefit of model compression.

Our work shows that knowledge distillation can bridge some of the performance gap between large and small models, making it possible to deploy lightweight models with confidence in scenarios where the full model is impractical. The ResNet18 student, guided by the ResNet34 teacher, was able to learn the essential features of the data and outperform what we would expect from training. This confirms findings from prior research that distilled models can retain a significant portion of a teacher's accuracy.

We also discussed the limitations: the student did not fully reach the teacher's performance, indicating room for improvement. For future work, we suggest exploring intermediate distillation (using teacher's hidden layer features), applying a teacher-assistant if the gap is large [6], and tuning hyperparameters like the temperature and loss weighting. Additionally, evaluating the method on a larger or more complex dataset would further demonstrate its scalability and effectiveness.

In conclusion, this project contributed a practical implementation and analysis of knowledge distillation in a computer vision context. It reinforces the idea that with the right techniques, we can compress neural networks and make them more efficient while still achieving strong performance. This has significant implications for deploying deep learning models on edge devices and for model compression research. Future research could apply our approach to other architectures and tasks and refine the distillation technique to further close the gap between compact models and their high-capacity teachers.

References

[1] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. NIPS Deep Learn. & Representation Learn. Workshop*, Montréal, QC, Canada, 12–13 Dec. 2014, arXiv:1503.02531.

[2] A. Romero *et al.*, "FitNets: hints for thin deep nets," *arXiv preprint* arXiv:1412.6550, Dec. 2014.

[3] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer," *arXiv preprint* arXiv:1612.03928, Dec. 2016.

[4] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: fast optimization, network minimization and transfer learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* (CVPR), Honolulu, HI, USA, 21–26 Jul. 2017, pp. 4133–4141.

[5] S. Srinivas and F. Fleuret, "Knowledge transfer with Jacobian matching," in *Proc. 35th Int. Conf. Mach. Learn.* (ICML), Stockholm, Sweden, 10–15 Jul. 2018, pp. 4723–4731.

[6] S.-I. Mirzadeh *et al.*, "Improved knowledge distillation via teacher assistant," in *Proc. AAAI Conf. Artif. Intell.* (AAAI), New York, NY, USA, 7–12 Feb. 2020, vol. 34, no. 4, pp. 5191–5198.