Understanding Teacher-Student Architectural Choices on Knowledge Distillation

Javier Roset, Davis Thomas, Vicente Lisboa

10th June 2023

Abstract

Knowledge distillation is an effective way to bring cutting-edge performance improvements in large over-parametrized models to smaller models. We aim to train a distilled student model using knowledge distillation from different SOTA teacher models and compare the performance of these models, as well as a baseline trained directly on the dataset. We also experiment with the tuning parameters to verify observations made by previous authors on this task. We find that distilled models perform better than models trained from scratch, and that the architecture and performance of the teacher models has an impact on the performance of the student.

1 Introduction

In recent years, there has been a surge in the development of large and complex deep learning models called "teacher" models. These models perform well in various tasks but require significant computational resources. To address this, researchers have explored knowledge distillation, a technique that transfers knowledge from a large teacher model to a smaller student model. This compression is useful for resource-constrained environments like mobile devices or IoT devices, enabling the deployment of efficient models without sacrificing accuracy.
Knowledge distillation allows the student model to leverage the information captured by the teacher model, gaining insights and patterns that are difficult to discover through traditional training methods. It also promotes model generalization by encouraging the student to learn from the teacher's soft predictions rather than relying solely on hard labels. This leads to improved performance, robustness against attacks, and resistance to input data noise.
In this paper, we aim to replicate the experiments conducted in [1] and explore the principles, methodologies, and advancements in knowledge distillation. We provide a comprehensive review of the existing literature, highlighting the benefits, challenges, and potential applications of this technique. Additionally, we conduct experiments to optimize the distillation process by considering architectural choices, loss functions, and temperature scaling.

2 Literature Review

Discussions on how to perform model compression to effectively transfer the performance of large ensembles of hundreds or thousands of base-level classifiers into smaller, faster models were detailed by the authors of [2], where they proposed a compression method by labeling a large unlabeled data set with the target model, and

then training a student model using the newly labeled data, or by generating pseudo data from a distribution similar to that of the true data when unlabeled data is not available. It is to be noted that the training objective here is to maximize the average log probability of the correct label, hence optimizing performance on the training data.
The term 'knowledge distillation' was first used by Hinton et. al. in [5], where they develop the approach proposed by the previous paper further using a different compression technique. The authors argue that the training objective should reflect the real objective, which is to generalize well to new data, which can be done by teaching a student model to generalize the same way the teacher does. They suggest a modification to the training technique that achieves this, by using the class probabilities produced by the teacher model as "soft targets" for training the small model. An explanation of the classical knowledge distillation techique as well as the one presented in [1], will be presented in the methodology section.
Finally Beyer et. al. in [1] explores the knowledge distillation task by framing it as a function-matching task. They conduct rigorous experiments to suggest the best techniques and approaches that should be used to make model distillation work, and suggest three main ingredients that deliver the best distillation performance:

1. The training has to be consistent. For training to be consistent, both the teacher and the student need to see identical inputs, including augmentations.

2. Introduce aggressive data augmentations to enrich the input image manifold (through mixup). We want to give as many support points as possible to allow the student to learn the function of the teacher.

3. Use very long training schedules.

3 Data

For our analysis we used the CIFAR-100 dataset, an extension of the CIFAR-10 dataset. This dataset is a widely used benchmark in the field of computer vision that enables fine-grained visual recognition tasks.

CIFAR-100 consists of 60,000 color images, each of size 32x32 pixels, distributed across 100 fine-grained classes. These classes are further grouped into 20 superclasses, forming a hierarchical structure that reflects the natural organization of visual concepts. Each image comes with a "fine" label, (the class to which it belongs) and a "coarse" label (the superclass to which it belongs). The dataset is carefully curated to ensure intra-class similarity and inter-class variability, enabling the evaluation of algorithms in challenging scenarios.

The dataset is split into two mutually exclusive subsets: a training set and a test set. The training set contains 50,000 images, with 500 images per class, while the test set consists of 10,000 images, with 100 images per class. The distribution of classes in both sets is designed to maintain the balance of samples across the entire dataset, allowing for fair evaluation of algorithms.

4 Methodology

4.1 Explanation of the technique used

4.1.1 Classical knowledge distillation approach

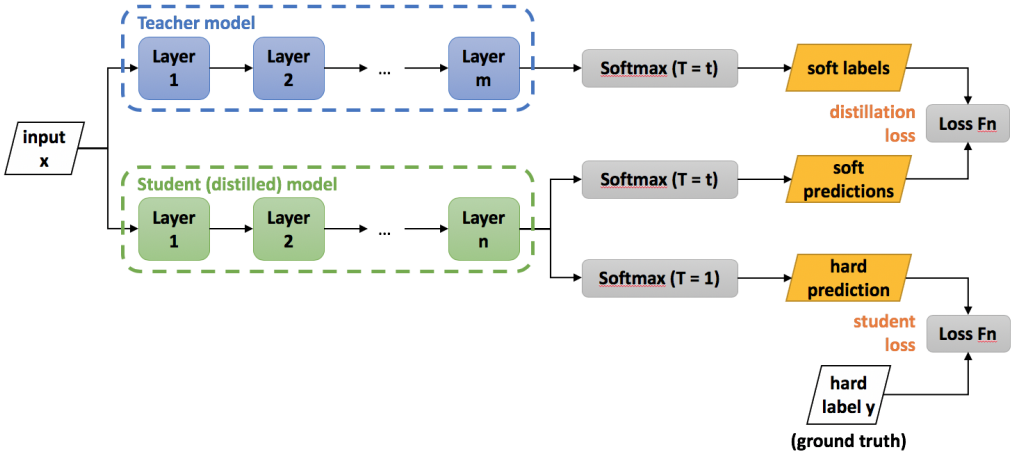


Figure 1: Diagram of knowledge distillation

In figure 1, we can see a representation of how knowledge distillation works. The main idea of knowledge distillation is the following; first, you input an image on both, the pre-trained teacher (big, more capable model) and the student (small and efficient model). Then you process the image for both and obtain the logits. With those logits we will compute 2 losses, one will calculate the distance between the hard labels (ground truth) and the outputs of the student, for this one a typical choice is categorical crossentropy. The other one, will calculate the distance between the outputs of the teacher and the student, with the objective of forcing the outputs of the student to be similar to the outputs of the teacher. To do so, one typical choice for this loss is Lullback Leibler divergence loss. Apart from that, we will use a softmax with temperature for this second loss. This softmax can be expressed as:

$$q_i = \frac{\exp\left(z_i/T\right)}{\sum_j \exp\left(z_j/T\right)}$$

Increasing this temperature results in a softer probability distribution, as can be seen in image 2, where we see how the relative difference between classes decreases when the temperature is increased.

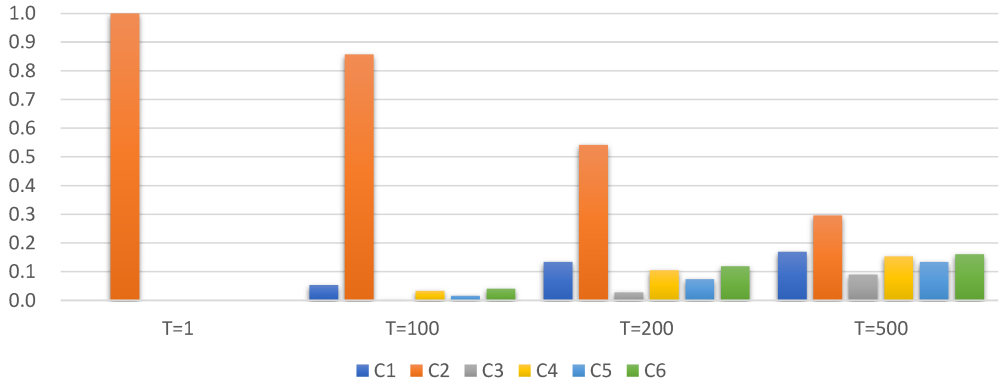


Figure 2: Effect of temperature in softmax

Conceptually what we are doing is the following. If we assume that the teacher has very high accuracy on the target dataset, it will output a probability distribution that will mostly represent the target class of that image, meaning that for example in an image of a dog, the output will be 70% for dog, and less than 5% for the rest of the classes. This is very similar to the hard label, meaning that it doesn't contain a lot of information from the image. But after the training, the teacher is capable of expressing a lot more information about the image, such as *"this image is most likely a dog, but it is also similar to a cat, and is completely different from a car"*. This information can be expressed with these soft labels, and by using the Kullback Leibler loss, can be transferred to the student.

4.1.2 *A good teacher is patient and consistent* approach

In this paper, the main changes they do to the knowledge distillation technique are the following:

- They **ONLY** use Kullback Leibler divergence loss, which means that they are not using labels for the training at any point. They only want the student to **match** the function the teacher has learned.
- They ensure that teacher and student see the same image at all times (consistency concept).
- They apply mixup, an aggressive data augmentation technique, to generate support points outside the original image so that the student has a larger representation of the teacher's function (function matching concept).
- Really long trainings (between $10^3 - 10^6$ epochs)

4.1.3 Mixup augmentation

Mixup consists of linearly interpolating 2 images in the way figure 3 describes. You get 2 images, set a lambda parameter to indicate the percentage of each image, and then modify the labels with the same percentages. That way you get an image that is partly dog and partly cat, and you ask the model how much of each class the image has. It is a very simple approach that can help in generalization.

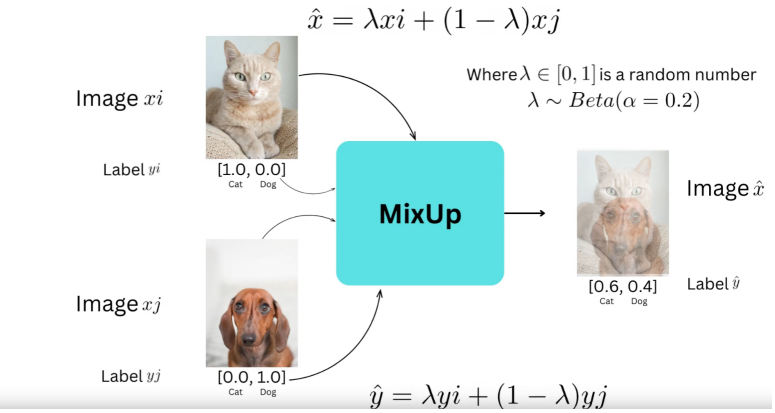


Figure 3: Mixup augmentation concept

Mathematical intuition of mixup

Imagine the N-dimensional space where the images stand (the space of natural images). What a deep learning model is going to try to learn is to which class every cluster of image points correspond as figure 4 suggests.

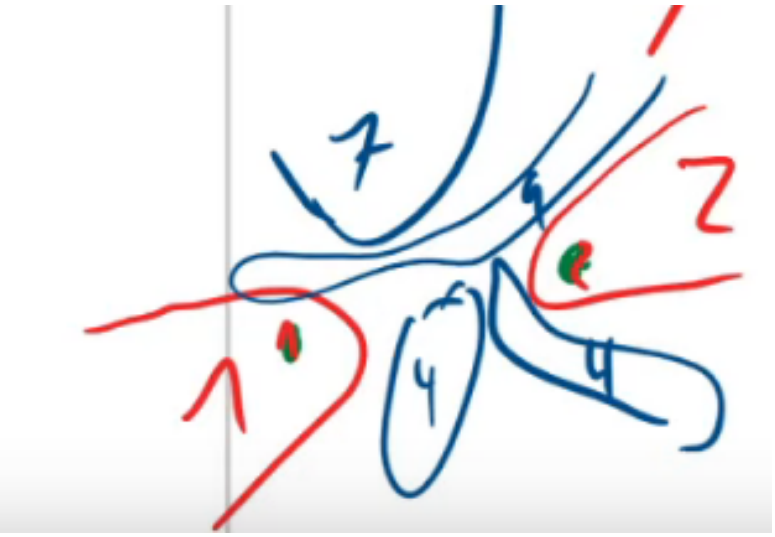


Figure 4: Representation of clusters of images in the space of natural images

The problem is that we only have a subsample of this space of natural images, so what we ask the model to solve is an empirical risk minimization problem[3] meaning that it will only focus on the space of the natural images that the training data comes from, but not the rest of it. To give a visual example, look at figure 4. The classes the model learn to detect occupy a space in the natural images space defined by the images the model has seen from the training set. But what happens to the space between those clusters? Mixup tries to use this unused space generating more support points for the model to learn by linearly interpolating images, generating data points for all the points in between two images, creating the isolines between 2 classes as expressed in 5. This helps the model to generalize, and also for this concrete application **gives more support points for the student to learn the function of the teacher**.

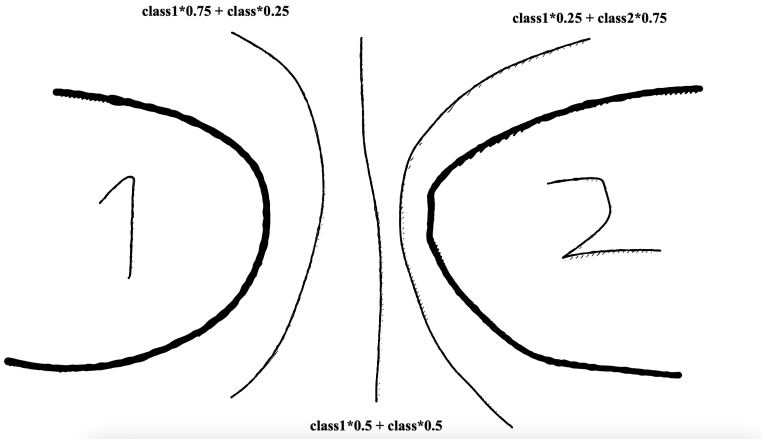


Figure 5: Mixup augmentation, mathematical intuition

5 Experimental Setup

Using a pytorch implementation, the training loop was defined such that the teacher and the student models see the same input data, and the predictions of the student are compared to the predictions of the teacher using the Kullblack-Leibler(KL) divergence loss. This ensures that the objective of the training is for the student to learn from the soft predictions of the teacher, rather than just the hard labels which would be the case for a cross-entropy loss on the predicted outputs. After each training epoch, the updated model is tested on the validation set and the metrics are recorded. The code for our experiments can be found in the linked repository

5.1 Model Choice

For the choice of the teachers, we selected models with different architectures, and different computational costs (MAdds) to compare their training performance. The metrics of these different teachers have been tabulated below. The student was chosen for its small size and lowest MAdds score. It is important to note that the scores in the table are from the pretrained models, and we train the same student from scratch for each experiment.

Ref	Model	Top-1 Acc.(%)	Top-5 Acc.(%)	#Params.(M)	#MAdds(M)
Teacher 1	ResNet44	71.63	91.58	0.67	97.44
Teacher 2	VGG11_bn	70.78	88.87	9.80	153.34
Teacher 3	MobileNetv2_x0_75	73.61	92.61	1.48	59.43
Student	MobileNet_v3_small	73.69	92.31	1.61	17.6

Table 1: Metrics of the selected models (pretrained)

5.2 Optimization

We used a cosine learning rate scheduler without restarts, so the learning rate would slowly decrease over the training epochs. For the student model we trained from scratch, we implemented an Early Stopping

callback to prevent overfitting on the training dataset. We used the Adam optimizer since it demonstrates good performance on a wide variety of applications, however we sweep over different initial learning rates during our experimentation. Another part of our strategy is the mixup augmentation, which we implemented in accordance with the recommendations of Beyer et. al.[1]

5.3 Summary of experiments conducted

.

1. Investigating the effect of different teacher architectures on the distillation process:
   We conduct three trainings for 400 epochs on the using the different teachers with an initial learning rate of 0.001 and keeping a constant temperature of 5.

2. Investigating the effect of different temperatures on the training process:
   We conducted three trainings with the MobileNetv2 teacher over the temperatures of 2, 5 and 10. This is done for 300 epochs.

3. Tuning the learning rate:
   At various steps of the experimentation, we vary the learning rate to determine the best optimal learning rate for our experiment. This is done for 300 epochs.

4. Investigating the effect of the mixup augmentation
   We train the student with and without mixup augmentation on the training data, and we compare the results. This is done for 300 epochs.

6 Results

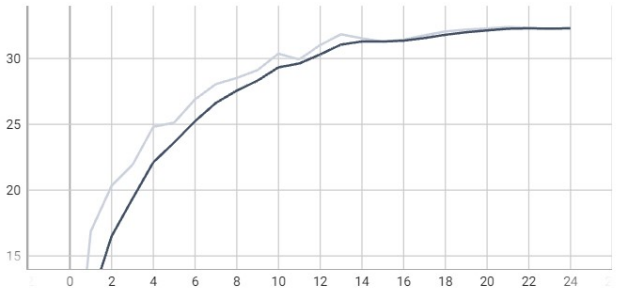
Teacher	Mixup	Learning Rate	Temperature	Top-1 Acc.(%)
Trained from scratch	N	0.001	NA	31.53
ResNet44	Y	0.001	5	40.67
VGG11_bn	Y	0.001	5	37.26
MobileNetv2_x0_75	Y	0.001	5	42.22
MobileNetv2_x0_75	Y	0.001	2	39.14
MobileNetv2_x0_75	Y	0.001	10	43.42
MobileNetv2_x0_75	Y	0.01	10	40.97
MobileNetv2_x0_75	Y	0.003	5	42.3
MobileNetv2_x0_75	N	0.001	10	48.21

Table 2: Accuracy comparisons of our trained models

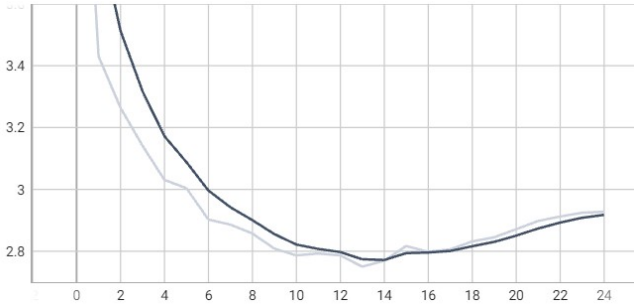
6.1 Baseline model trained from scratch

This model was trained using the same student but with an early stopping callback implemented to prevent overfitting on the training dataset. Mixup augmentation was not implemented, and we used the cosine annealing

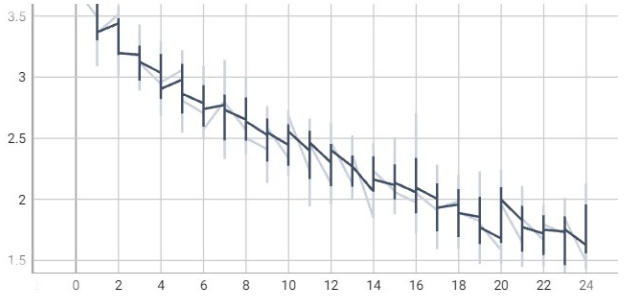
scheduler for tuning the learning rate. From the table?? we can observe that all of our distilled models have been able to beat the baseline. There are pretrained SOTA models on pytorch with a higher accuracy than our baseline, however these were pretrained on much larger ImageNet datasets, before being finetuned on CIFAR-100.



(a) Accuracy over validation



(b) Loss over validation

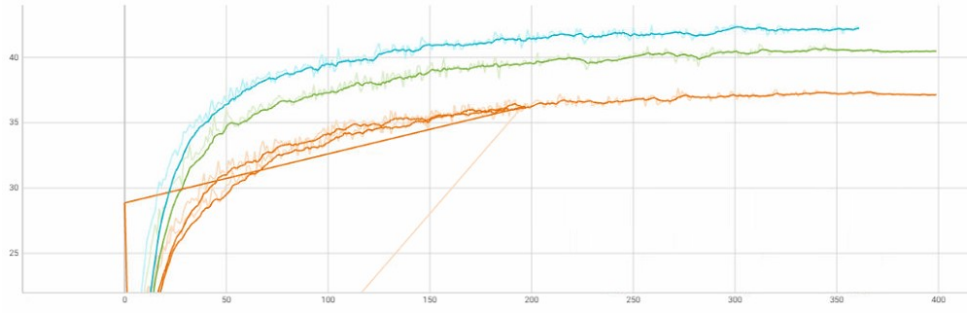


(c) loss over training

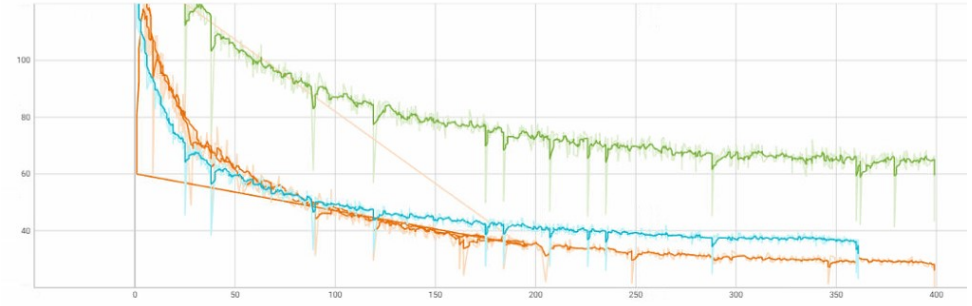
Figure 6: Metrics of model trained from scratch

6.2 Effect of teacher-student architectures

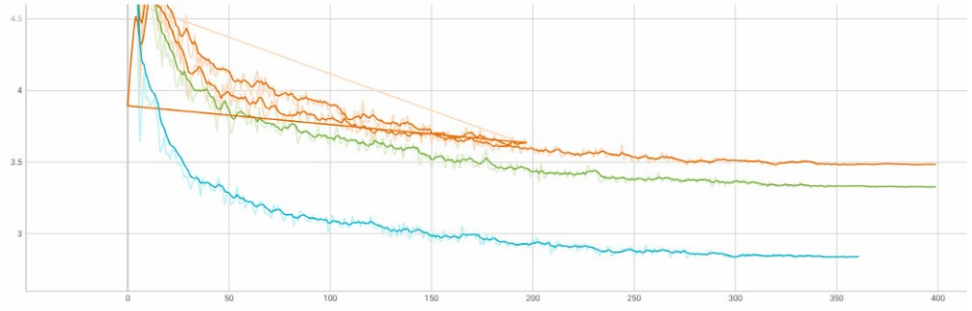
We can see from Fig.7 that the choice of teacher-student architectures has a significant impact on the knowledge distillation process. The model that exhibits the highest performance and shares a closer architecture to the student achieves the highest training accuracy in knowledge distillation. This suggests that a teacher model with a similar architecture to the student facilitates better knowledge transfer, potentially due to a closer alignment of representation spaces and feature extraction capabilities. This is in line with the findings of Gao et. al [4], where the authors propose that degradation of knowledge transfer can occur if there is a model capacity gap between the large deep neural network and a small student neural network.
These results highlight the importance of carefully selecting teacher models that are compatible with the student architecture for effective knowledge distillation.



(a) Accuracy vs number of epochs over the validation set



(b) Loss over training



(c) Loss over validation

(d) Accuracy of teacher vs number of epochs over the training set

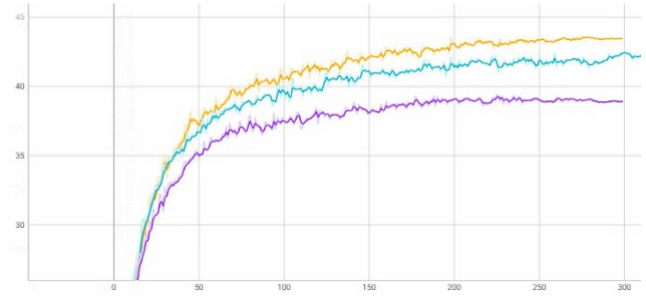
: Teacher 1 : Teacher 2 : Teacher 3

Figure 7: Metrics with different teachers vs number of epochs

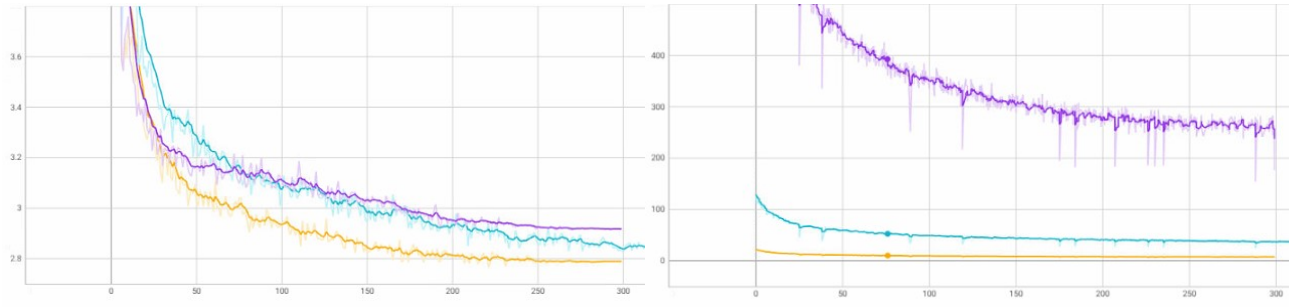
6.3 Hyperparameter exploration

From the results of our experimentation with the hyperparameters we can observe the following:

- From our experiments sweeping over the distillation temperature, over values 2,5 and 10, in the fig 8 we can see that a temperature of T=10 has the highest accuracy, and this is evident from the loss curves where we can see that the lowest loss over validation is observed in this case. This could be due to the fact that scaling the temperature leads to softer probability distributions, allowing the student model to capture more nuanced information from the teacher.



(a) Accuracy over validation



(b) Loss over validation

(c) Loss over training

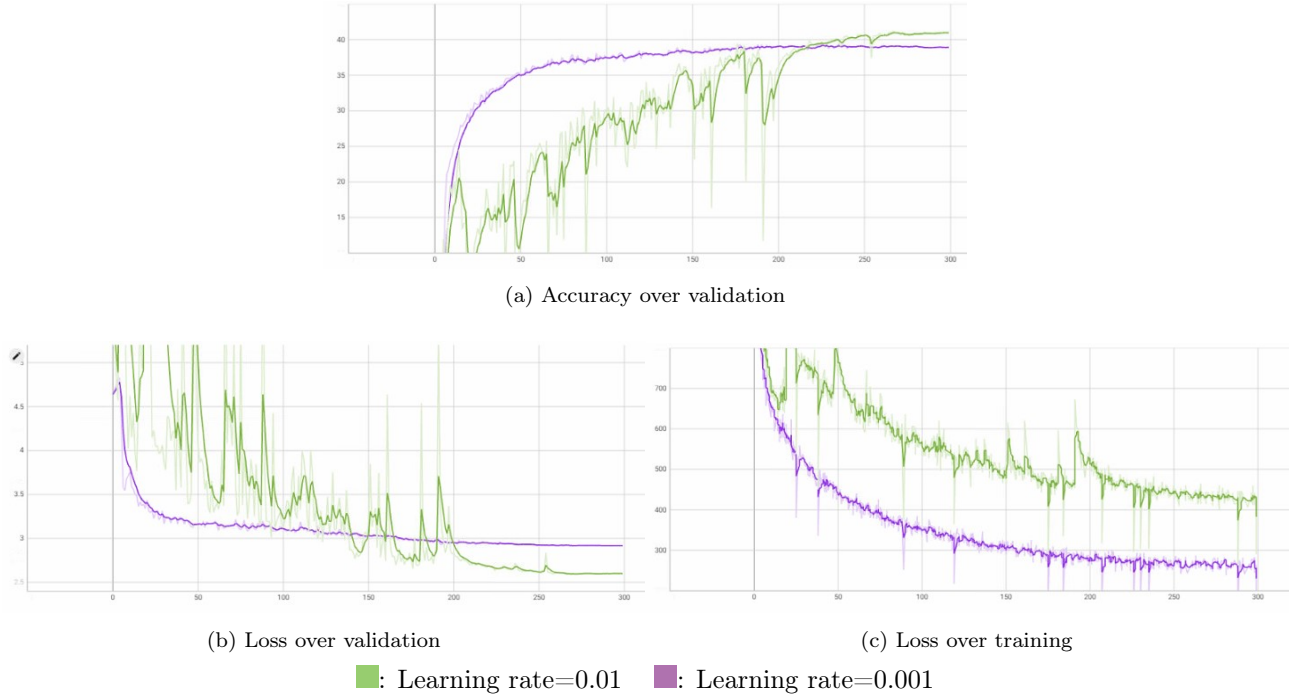
: T=10 : T=2 : T=5

Figure 8: Metrics of tuning temperature hyperparameter

- Due to time and resource constraints, we were not able to sweep over the learning rates while keeping the temperature constant. However from the results comparing the effect of initializing the learning rate at 0.01 or 0.001 at temperature T2, from 9 we can see that 0.01 is very noisy in its convergence, even though it ends with a higher accuracy. Similarly from 10, where we vary the learning rate between 0.001 and 0.03, we can observe similar noise in the training with the higher value.

6.4 Investigating mixup augmentation

An ablation study of the mixup augmentation reveals that increased accuracy is observed with mixup removed. This is counter to the recommendations of the authors in [5], however we hypothesize that the teacher model also might need to be trained using mixup augmentation, or that the resolution of the images is too low to benefit from this technique. Finally, we observed that longer training schedules resulted in better performance



(a) Accuracy over validation

(b) Loss over validation

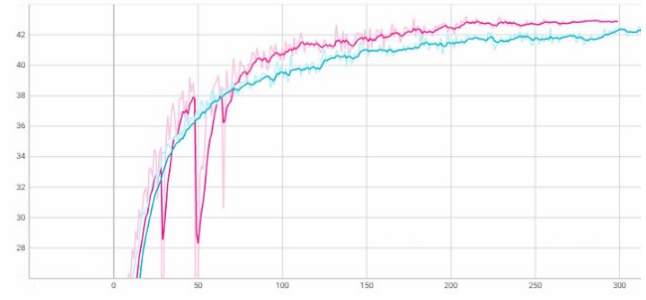
(c) Loss over training

Figure 9: Metrics of tuning learning rate with constant temperature of T=2

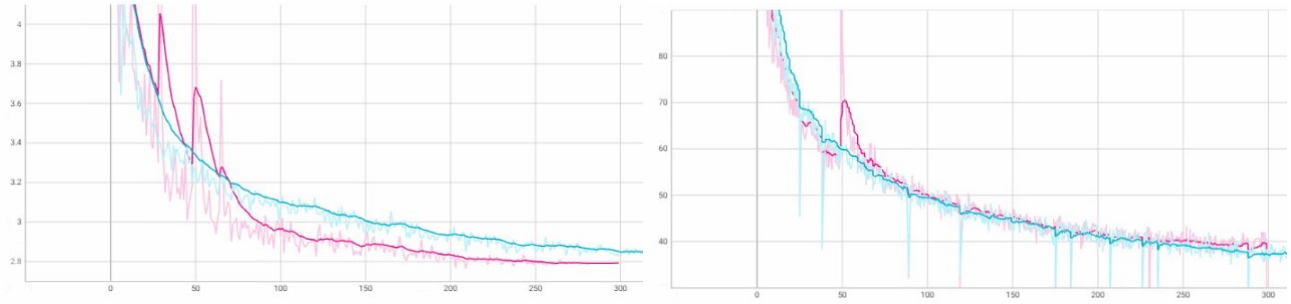
for the student models. Extending the training duration allowed the models to converge to better optima and refine their representations.

7 Conclusion

Overall, our experiments demonstrated the efficacy of knowledge distillation in transferring knowledge from teacher models to student models. The choice of teacher model, distillation technique, and hyperparameters significantly influenced the student's performance, highlighting the importance of careful selection and tuning in the distillation process. Although we aimed to replicate the results of the paper[1], achieving the same level of accuracy with a smaller model, we were unable to fully accomplish this objective. Potential reasons for this include not training for long enough periods and not considering other conditions necessary for the technique to work effectively, such as requiring images with a higher resolution.
Further exploration and refinement of these factors are necessary to fully leverage the potential of knowledge distillation in model compression and knowledge transfer tasks.



(a) Accuracy over validation

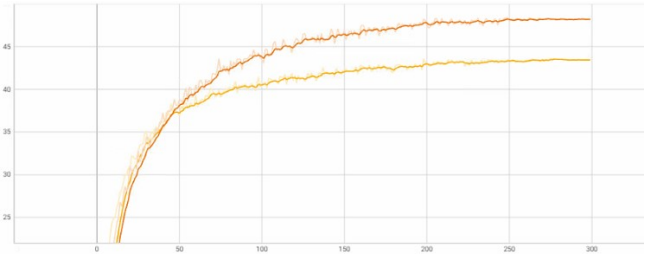


(b) Loss over validation

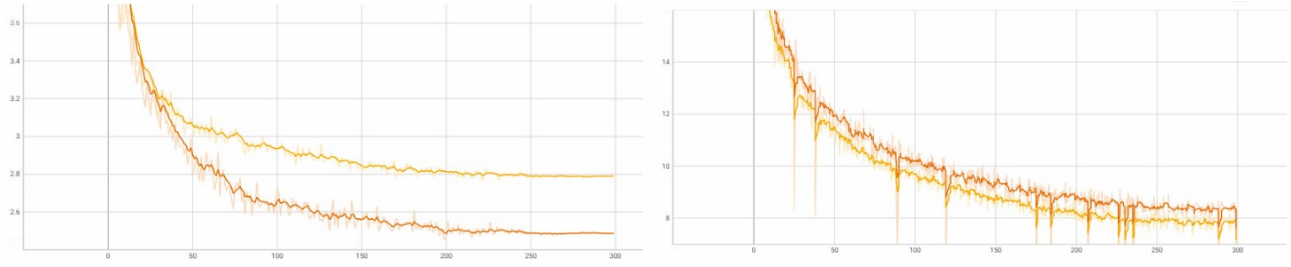
(c) Loss over training

: Learning rate= 0.003 : Learning rate= 0.001

Figure 10: Metrics of tuning learning rate with constant temperature of T=5



(a) Accuracy over validation



(b) Loss over validation

(c) Loss over training

: Without Mixup : With Mixup

Figure 11: Metrics comparing the effect of mixup augmentation

References

[1] Lucas Beyer et al. "Knowledge distillation: A good teacher is patient and consistent". In: *CoRR* abs/2106.05237 (2021). arXiv: 2106.05237. URL: https://arxiv.org/abs/2106.05237.

[2] Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. "Model compression". In: *Knowledge Discovery and Data Mining*. 2006.

[3] *Empirical risk minimization*. URL: https://en.wikipedia.org/wiki/Empirical_risk_minimization. (accessed: 9.06.2023).

[4] Mengya Gao et al. *Residual Knowledge Distillation*. 2020. arXiv: 2002.09168 [cs.LG].

[5] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. 2015. arXiv: 1503.02531 [stat.ML].