Project A

ECE 1512 Digital Image Processing and Application

Due Date: Nov.1, 2023

Lanzhi Xiao
uid:1004569362

Bocheng Zhang
uid:1009969517

Introduction

Project A consists of two tasks. The first task delves into the technique of Knowledge Distillation (KD). KD is a popular method for condensing models where a more compact model, termed the "student", is trained using insights from a more robust "teacher" model. The soft target from the teacher model would give more information to the student model for its training. In implementing the state-of-the-art part, we explore the subclass Knowledge Distillation, we explore the meaningful ways to split, and also different loss functions which allow the student model to learn more details. In the second task, we treat the ResNet50V2 as the teacher model, MobileBetV2 as the student model. For the initial training, we freeze the fine-tuning classification layer to do the feature extraction for further training, then we add the fine-tuning whole process for the 25 epochs. Finally, we modify our code to separate more subclasses based on the number of annotators who selected SSA, and assign different weights to loss for the how big of the difference from teacher to the student model. As a result, we learn the performance improvements using different techniques.

1 Task 1

1.1 Question Answer

a. Knowledge Distillation (KD) is compression technique used to compress the knowledge in an ensemble into a single mode in the paper. According to Rich Caruanna and his collaborators (2006) [1], once the teacher model has been trained, we can use distillation to transfer the knowledge (KD) from teacher model to student model. The student mode, a small model, can use knowledge distillation to gain the generalization ability from the teacher model. Consequently, the student model acquires a comparable proficiency to that of the teacher model, bypassing the need for extensive training data.

b. The knowledge that the teacher model transfers to the student model is the form of the soft targets. So for tasks like MNIST, the teacher model can produce the correct answer with high confidence, and the information can tell the correct answer to the different class symbols, which can be used by the student model. For example, if we only use the hard targets to do the classification problem with three classes, a hard label for an image of cat would be [1,0,0] indicating the image belongs to the cat class. However, if we use the soft targets which might be [0.8, 0.1,0.1], the student model can learn that the image is not like a dog and not like a bird in the same level, so the model can capture this uncertainty and make more nuanced predictions.

c. The temperature parameter T is a crucial element in Knowledge Distillation (KD) as it governs the "softness" of the probability outputs from the teacher model. A lower T value leads to a sharper focus on the higher-probability labels, minimizing the influence of less likely ones. Conversely, a higher temperature setting allows for a more detailed representation of the information, including nuances, thereby enhancing the student model's capacity for generalization. Typically, a lower temperature is employed when the teacher model's predictions are made with high certainty, reducing uncertainty, while a higher temperature is preferable in situations with greater data ambiguity to capture a broader spectrum of information.

d. Geoffrey Hinto and his colleagues [2] mentions that neural networks typically produce class probabilities by using a "softmax" output layer. The equation below to compute for each class into a probability, q_i:

$$q_i = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}} \tag{1}$$

In the equation T is a temperature and it is normally set to 1. The same high temperature is used when training the distilled model. The better way to produce the data that makes sure to use the correct labels to modify the soft targets is simply use a weighted average of two different objective functions [2]. The first objective function is to use cross entropy loss with the soft target, with the same high temperature in the softmax of the student model and teacher model.

$$L_{teacher} = -\sum_i p_i log(q_i) \tag{2}$$

$$L_{student\_target} = -\sum_i p_i log(q_i) \tag{3}$$

The function 2 and 3 have the same high temperature in the softmax, with T = 1. Meanwhile the second cross entropy loss of the student model with the correct label.

$$L_{student\_label} = -\sum_i p_i log(q_i) \tag{4}$$

We need to multiply $L_{student\_target}$ and $L_{student\_target}$ by T^2, because the magnitudes of the gradients produced by the soft targets scale as $1/T^2$ (Geoffrey and his colleagues, 2015) [2]. So the loss function of student will be:

$$L_{student} = \alpha L_{studen\_targe} + (1 - \alpha) T^2 L_{student\_label} \tag{5}$$

The task balance parameter would decide which loss would have a larger impact on the student model. When the task parameter is large in function (5), the student model would be influenced more by its own hard target, comparatively, if the task balance parameter is small, the soft targets from the teacher model would influence the student more, we need to do lots of trails-and-errors to set the best parameters.

e. Regularization is a technique used to prevent overfitting in the machine learning process. So we can regard the KD as a regularization technique, since in KD, we use soft targets produced by a larger, more complex model to regularize the training of a smaller, simpler model. As Geoffrey[2] said in this paper, " the soft targets produced by the teacher model can be viewed as a form of regularization that encourages the student model to learn a smoother decision boundary that generalizes better to new examples." The soft targets provide additional

information to the student model that is not present in the true labels, this additional information which can be viewed as the extra information which can help improve the generalization performance. Then this approach can help prevent overfitting.

1.2 Data Load

For the data loading process, we used the MNIST dataset, we built the teacher and student model based on the diagram. Since we have 0-9 digits, our final NUM_CLASSES should also be 10.

1.3 Loss Function Implement

In this part, we implement the loss function for both teacher model and student model, so that our model would predict the probability and also the loss for each label. Specifically, the teacher loss function would produce the softmax loss for each label, and calculate the average loss for all datasets in the batch. The similar process with the distillation loss for the student model but with extra temperature, which controls the softness of the probability distributions produced by the teacher model. Then, the compute student loss function combines the soft target loss from the teacher model, and hard target loss from the true labels, the Alpha is used to balance the influence for both.

1.4 Train and evaluation

For different values of temperature and alpha, we have tried [1,2,4,16,32,64] for T, and [0.2,0.5,0.8] for alpha. The way we implement the comparison is using the loop to search like all the combinations. The outcome is for the student model with KD, the model with the best class_accuracy has alpha = 0.5, T= 4.

1.5 Teacher Model and Student Model Training and comparison

The aim of this part was to train both the teacher and student models over 12 epochs and subsequently evaluate their performance on test data. The temperature for the student model (fc_model) is 4 and alpha value is 0.5. For each epoch, the teacher loss was computed for the teacher model and the student loss was computed for student models. As a result of the training, the accuracy for the teacher model is 99.31% and the accuracy for the student model is 98.79%. The teacher model outperformed the student model by 0.52%.This experiment demonstrated the effectiveness of knowledge distillation (KD) from the teacher model to the student model under the given parameters.

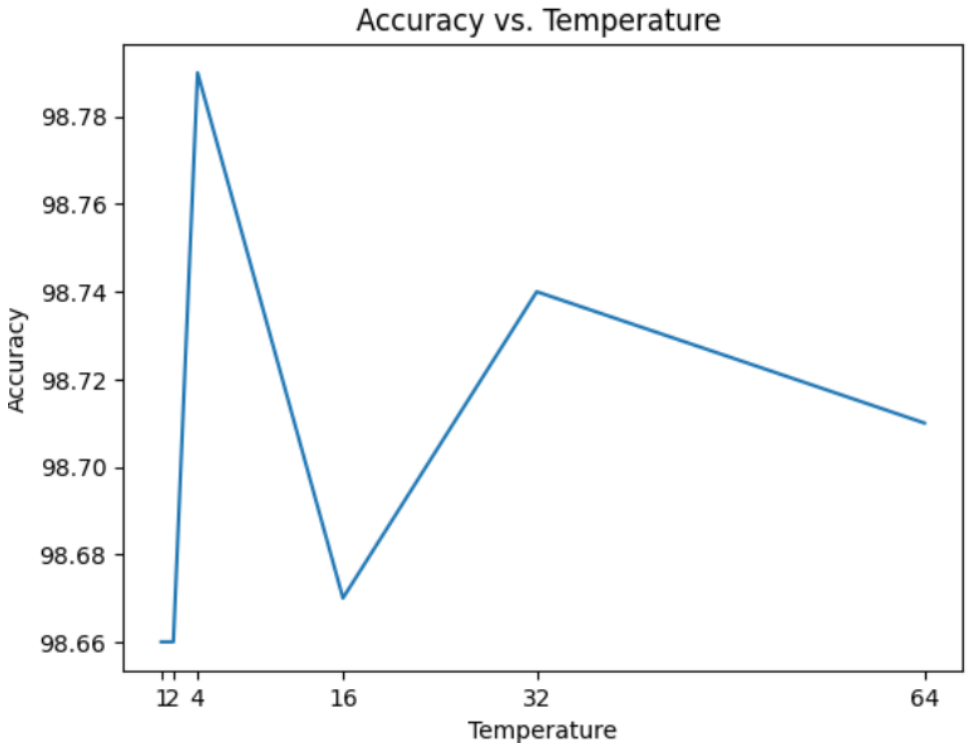
Model	Test Accuracy
Teacher Model	99.31%
Student Model with KD	98.79%

Table 1 : Test accuracy for teacher model and student model with KD

1.6 Temperature Hyperparameters Experiment

In this part, testing the impact of temperature hyperparameters on student test accuracy with the same ALPHA value (0.5). The temperature hyperparameters will be setted by plotting a curve representing student test accuracy vs. temperature hyperparameters (T = 1,2,4,16,32, and 64. In Plot 6.1, the x-axis

is the temperature value and y-axis is the accuracy. In the experiment, the epoch will be 12. From the Plot 6.1, the highest accuracy is 98.79%, with the temperature (T =4) and the lowest accuracy is 98.66%,with the temperature (T=1). Judging from the curve trend, too high or too low temperature will affect the accuracy of the student model. The T value helps the student model learn generalization information from the teacher model, compared with T =1, the teacher model would pay more attention to the negative labels which means the student model could learn more generalization and more cases.



Plot 6.1: plot of student test accuracy and different temperature hyperparameters

1.7 Student Model Without Knowledge Distillation (KD)

The student model trained with knowledge distillation (KD) tends to have better accuracy than the student model trained without KD, with the teacher model having high accuracy, 99.17%. The accuracy value of the student model trained with KD is 98.79%, with temperature 4, and the accuracy value of the student model trained without KD is 98.10%. For the student model with KD, there is a teacher model that provides additional guidance to the student model by teaching it about labels.

Model	Test Accuracy
Student model with KD	98.79%
Student model from scratch	98.11%

1.8 Compared Model by Floating Point Operations Per Seconde (FlOPs)

The FLOPs (Floating Point Operations Per Second) refers to the number of floating-point calculations a computer or processor can perform in one second. In order to calculate the FlOPs of a model, we use

the get_flops function from Tokusumi [9]. The table below shows the FLOPs about the teacher model, the student model with KD and the student model without KD. From the table, the teacher model with the highest FLOPs value, 22042058 and the student model with KD or the student model without with the same FLOPs value, 2475882. Thus, Knowledge distillation does not change the computational complexity (FLOPs) of the student model, as the structure of the student model remains the same whether KD is applied or not. Meanwhile, the teacher model is more computationally intensive than the student models.

FLOPs of Teacher Model:	22042058
FLOPs of Student Model With KD:	2475882
FLOPs of Student Model Without KD:	2475882

Table 2.8: FLOP Result for 3 Models

(Student Model/ Teacher Model = 2475882/ 22044058 = 11.23%, which means Students' FLOPs are only 11.23% of teachers' FLOPs)

Additionally, there are 9 layers in the teacher model (CNN) and the total trainable parameter, weights and biases that the model learns during the training process, is 1011466 (3.86 MB). The student model with KD and the student model without KD have the same neural network architectures, 4 layers, and the training process is 1238730 (4.73 MB) for both models.

1.9.1 Novel Idea of Paper No. 15

The main novel idea of this paper is the Subclass Knowledge Distillation(SKD) framework from Ahmad [7], which efficiently distilled subclass knowledge into the student network to improve tts performance in classification tasks. The way that method can achieve such improvements is that the SKD process leverages information on existing subclasses within each class to convey meaningful information on existing subclasses within each class to convey meaningful information which is not in the teacher's class logits but exists in subclass logits. For example, if the teacher model has only a few classes, the amount of information transferred from the teacher to the student is restricted, the class labels might only be "fish", "birds", if SKD can be applied , the subclass of "fish" might be extra information which can be used to boost the student model. Furthermore, the majority of training samples are identified as normal class, resulting in a biased dataset. In this case, the SKD framework can further categorize the small differences within the large class, which can be extra details.

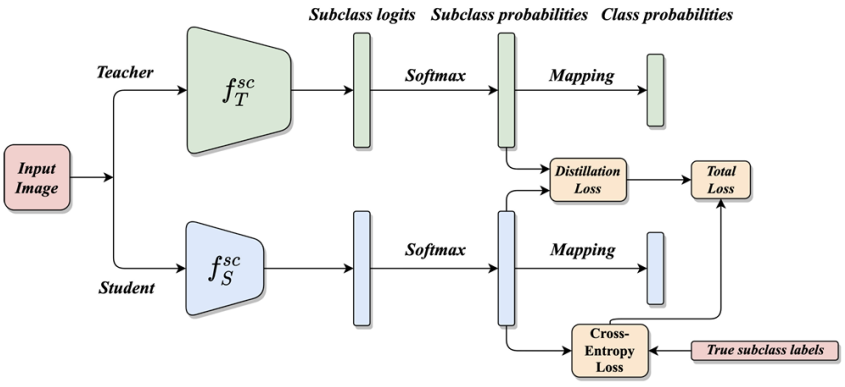


Fig. 1. Subclass Knowledge Distillation framework.

1.10.1 Propose Method Improve Student Performance of Paper No.15

The Subclass Knowledge Distillation (SKD) technique outperforms traditional Knowledge Distillation (KD) by passing on subclass predictions from a larger teacher network to a smaller student network, enhancing the latter's classification capabilities. SKD utilizes subclass data within each class for better results. In a clinical study for colorectal polyp binary classification mentioned in paper [3], a student model trained using SKD achieved an 85.05% F1-score, marking a 1.47% increase over traditional KD training and a 2.10% improvement compared to training without KD.

1.11.1 Limitations of the Proposed Method of Paper No.15

The SKD assumes that there are a limited number of subclasses with each class, and when we set the temperature to be high, the information transferred from the teacher model to the student can be linear. If there are too many subclasses, the method may not be effective in conveying meaningful information. Possibly, assuming a limited number of subclasses, a hierarchical classification approach can be used to effectively convey meaningful information to the student network. Plus, the SKD framework requires the availability of subclass labels, which may not always be feasible or practical in real-world applications. This can limit the applicability of the method in certain scenarios. Using machine learning such as K-means to preprocess the data to separate the subclass may be feasible.

1.12.1 Implementation of Proposed Method of Paper No.15

Based on the idea of the SKD, we need to split more subclasses and do the implementation. We divide the dataset to be expanded to 4 times its original number of classes. Next, I change the model for student and teacher, since the teacher model needs to predict more subclasses based on its inputs. Then, the teacher model loss function we add the auxiliary loss.
According to the methodologies outlined in research paper [7], we have embarked on a nuanced approach to refining our classification capabilities. The initial phase involved transitioning from a rudimentary 10-class system to an elaborate structure encompassing 80 subclasses. This transition is motivated by our intention to extract intricate patterns and subtleties from the data. To align with this granularity in the data, our model architecture has been duly modified to accommodate the proliferation of subclasses. In terms of loss computation for the teacher model, our approach is two fold. The primary loss metric utilized is the conventional softmax cross-entropy loss. Complementing this is an auxiliary loss. This auxiliary loss is meticulously crafted to assess the similarities between normalized logits yielded by the model. To put it succinctly, logits with pronounced similarities result in a diminished loss, whereas those with stark differences culminate in an augmented loss. This correlation is achieved through the computation of dot products between normalized logits and their respective transposed versions. The resulting matrix serves as a repository of cosine similarities between logits, capturing their relationships for every pair of images within the batch.While the outcome of this rigorous training regimen mirrors the original in many respects, a significant elevation in accuracy remains elusive. One plausible reason could be the already stellar accuracy metrics exhibited by the initial classification system, rendering substantial improvements challenging. Additionally, the multifaceted interplay between the number of subclasses, temperature, and the alpha hyperparameters might not have been optimized to its fullest potential. Despite experimenting with a plethora of configurations, we believe that with more rigorous fine-tuning and trials, there's scope for the model to transcend its current accuracy thresholds, potentially achieving results that are markedly superior.

Model	Test Accuracy
Student with KD	98.79%
Student with SKD	98.82%

1.9.2 Novel Idea of Paper No. 9

The main novel idea of “Improved Knowledge Distillation via Teacher Assistant" [4] is that using the teacher assistant model to train the student model when there exists a gap between the teacher model and the student model. In the article, Seyed Iman Mirzadeh’s group mentioned that if the gap between students and teachers is large, the performance of the student network will be decreased. In order to fix this problem, there is a teacher assistant (TA) model (intermediate-sized network) to be implemented to bridge the gap between the student model and the teacher model. Meanwhile, in the article, they study the effect of teacher assistant size and extend the framework to multi-step distillation.

1.10.2 Propose Method Improve Student Performance of Paper No.9

In the article [4], Mirzadeh’s group mentioned that if the gap (in size) between teacher and student is large, knowledge distillation will not be effective. In order to overcome the gap problem, they created a new distillation framework called Teacher Assistant Knowledge Distillation (TAKD). In order words, there is an intermediate model, called teacher assistants (TAs) model, between the teacher model and the student model to fill in their gap. In the end of article [4], they create a student model, with ResNet8 and ResNet4 to test the improvement of the TAKD framework. The accuracy for the student model (ResNet8) using the TAKD framework is 88,01 and the accuracy for the student model (ResNet4) using the TAKD framework is 91.32%. By comparing the baseline knowledge distillation (BLKD) of two student models, ResNet8 (86.66%) and ResNet (89.75%), the Teacher Assistant knowledge Distillation to effectively remedy the gap problem.

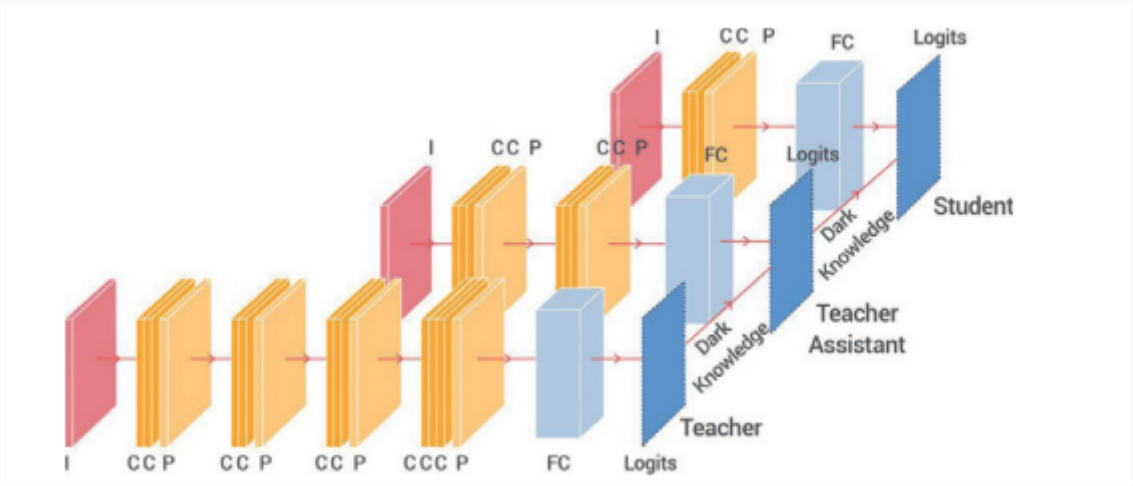


Figure 1.10.2: TA fills the gap between student & teacher [4]

1.11.2 Limitations of the Proposed Method of Paper No.9

The teacher assistants knowledge distillation (TAKD) can efficiently improve the knowledge distillation. However, there is a limitation for the number of teacher assistants, the best path given to constraints and time and computing resources. For the limitation of the best path given to contractions, Mirzadeh's team [4] mentioned that if there are n possible intermediate network, then there are will have 2^n possible path. So, it is very hard to find the best path if there is a gap between the teacher model and the student model. Additionally, there is a limitation of time and computing resources. Training a model requires time and computing resources, so increasing the number of intermediate models will take more time and computing resources to train. Thus, the interesting extension is improve the knowledge distillation by TAKD with limit the number of TA models.

1.12.2 Implementation of Proposed Method of Paper No.9

According to the article "Improved knowledge distillation via teacher assistant" [4], TAKD framework efficiency improves the knowledge distillation. In this part, we implement an TA model as a bridge for the teacher model and the student model. The teacher model is a CNN model with 9 layers. The student model has a lighter size than the teacher model, 4 layers. So, we create a TA model with 6 layers (mid number between the teacher model and the student model). The loss function for teacher model is standard cross-entropy: The loss for the student model and the TA model will be below:

$$L_{kd} = \tau^2 KL(y_s, y_t) \quad (6)$$

$$L_{student/ta} = (1 - \lambda)L_{sL} + \lambda L_{kd} \quad (7)$$

In the result, the accuracy for the **teacher model is 99.27%**, the accuracy for the **TA model is 99.13%** and the accuracy for the **student model is 98.76%**. From the accuracy, the TAs have better performance than the student, so the gap between the teacher model and the student model will influence the KD. Meanwhile, another model uses the baseline KD (directly studied from the teacher model), called BLKD. The accuracy for the **BLKD model is 98.70%**. Thus, the TAKD is an effective way to improve the accuracy of the student model.

1.13 XAI Methodologies Implement

In the exploration of model interpretability using Explainable Artificial Intelligence (XAI) methodologies, LIME (Local Interpretable Model-agnostic Explanations) plays a crucial role in elucidating the decision-making mechanisms of machine learning models.

In Figure 0, the visualization presents a comparative analysis of model attentiveness and decision rationale across different model architectures: a teacher model, a student model with Knowledge Distillation (KD), and a student model without KD.

The red regions in each explanation map denote areas of heightened significance, influencing the respective model's predictive outcomes. These critical regions are instrumental in the models' decision-making processes, guiding the attribution of class labels or continuous outputs.

A discernible observation is the scattered focus exhibited by the student model without KD, indicative of a less cohesive and potentially less reliable decision-making process. This dispersion may compromise the model's capacity to generalize effectively, possibly leading to suboptimal performance on unseen data.

Contrastingly, the application of KD in the student model appears to foster a more aligned and focused attention mechanism, mirroring the teacher model's decisive regions more closely. This mimicry suggests that KD facilitates a more refined knowledge transfer, enabling the student model to hone in on more pertinent features and attributes.

Consequently, KD's incorporation seems instrumental in enhancing the student model's explanatory coherence, steering its focus towards regions that are more critical for accurate and reliable predictions, thereby bolstering its generalization capabilities.

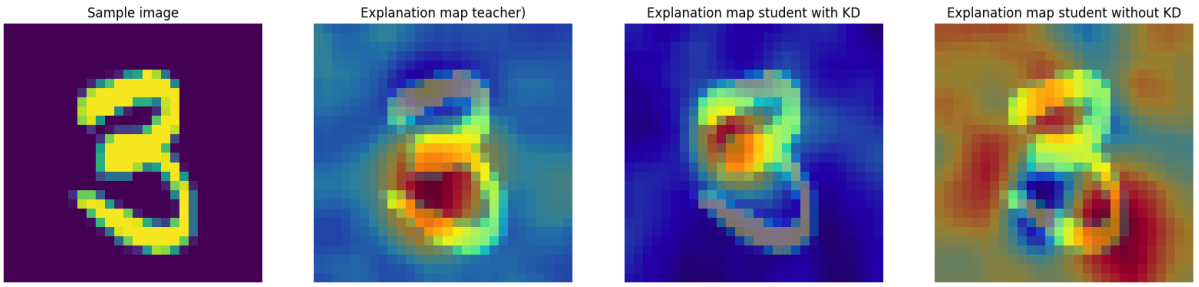


Figure0: the areas where the model is focusing when making the predictions

In order to achieve the LIME (Local Interpretable Model-agnostic Explanations), a methodology in Explainable Artificial Intelligence (XAI), we use the code from Mohammadi and Seyed[10].

2 Task 2

2.1 Question Answer

a. Transfer Learning is a process which involves leveraging a pre_trained model for a new, more specific work. For the Feature Extraction, the ResNet50V2 would extract meaningful features from MHIST Dataset,the process freezes the pre_trained feature extraction parts, then adds a few linear layers to do the final classifier. Compared with building the model from scratch, the training is usually faster and requires less data. The fine-tuning method unfreezes the backpropagation process of not only the last classifier, but also some of the pre-trained layers. For this project, we loaded the pre_trained ResNet50V2 as the teacher model and MobileNetV2 as the student model. They are all trained in a large dataset "ImageNet", we can add an extra classifier layer based on them to suit our needs to classify on MHIST, also we can choose to unfreeze more layers to allow the model to adjust its more abstract features for our specific task.

b. The residual block in ResNet architectures is stackable build blocks of the same connecting shape. Meanwhile, the central idea of ResNets is to use the additive residual function with a key choice of using an identity mapping. The purpose of Kaiming He's team is to propose a new residual unit, which makes training easier and improves generalization [5]. The equation shown below:

$$x_{l+1} = f(h(x_l) + F(x_l, w_l)) \quad [8]$$

In equation 8, F is a residual function. and f is a ReLU function, $h(x_l)$ is an identity mapping.

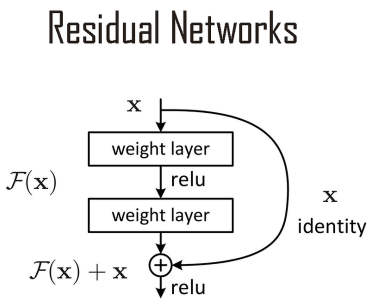


Figure1 : structure of Residual Block in ResNet[8]

c. The major difference between the ResNet-V1 and ResNet-V2 is the post-activation system as shown by [5].ResNetV2 incorporates an innovative design for its residual units, opting for a pre-activation method. Here, activation mechanisms like ReLU and BN are executed prior to the introduction of weight layers. This is a departure from ResNetV1's traditional post-activation system. Adopting this pre-activation technique facilitates enhanced gradient flow and regularization, subsequently optimizing the learning capabilities and adaptability of ResNetV2.

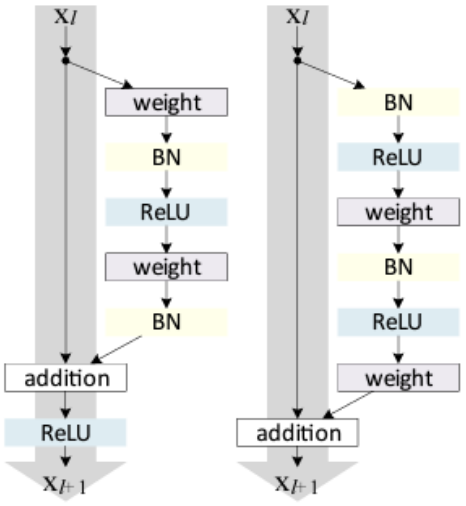


Figure2: structure differences between ResNet-V1 and ResNet-V2[5]

In addition ResNet-V1 and ResNet-V2 have different identity skip connection equation below:

- ResNet-V1:

$$X_{l+1} = X_l + F(X_l, W_l)$$

- ResNet-V2:

$$h(X_l) = \lambda_l X_l$$

The identity skip connection equation for ResNet-V2, which is simpler than the identity skip connection equation for ResNet-V1. In order words, the shortcut connections skip over two or more layers and directly add the input to the output.

d. MobileNetV2 advances from its predecessor, MobileNetV1, by integrating several enhancements. Firstly, it adopts an innovative inverted residual mechanism paired with linear bottleneck layers, enhancing the network's information flow. Secondly, a unique linear bottleneck layer is introduced, minimizing the network's parameters while retaining its precision. Next, the model incorporates the ReLU6 activation function, which offers better resilience in low-precision operations. Furthermore, the "squeeze-and-excitation" training method is introduced, optimizing the network's efficiency on limited datasets. Lastly, it presents SSDLite, a cutting-edge framework for object detection, surpassing the efficiency and accuracy of earlier techniques.

e. The vanishing gradient issue arises when training deep neural networks, especially those reliant on gradient-driven techniques such as gradient descent. This problem is characterized by the gradients of the loss function concerning the network's parameters, typically the weights, diminishing to near-zero values. ResNet architectures overcome this problem by using skip connections that allow for the direct flow of gradients connected to the earlier layers.The direct flow of gradients can preserve the information from the input, which decreases the possibility of losing information in the process.

f. MobileNetV2 is a lightweight model. Firstly, the basic building block of MobileNetV2 is a bottleneck depth-separable convolution with residuals. Mark Sandler and his colleagues [6] mentioned that the max number of channels/memory of MobileNetV2, 400K, is smaller than other models like MobileNetV1 (1600K) and ShuffleNet (600K). In addition, MobileNetV2 uses inverted residual bottleneck layers, which helps memory efficient implementation. Specifically, the total amount of memory will be dominated by the size of bottleneck tensors when treating bottleneck residual blocks as a single operation. MobileNetV2 has significantly fewer parameters than larger models. Fewer parameters mean less memory usage and faster computation, making it suitable for mobile devices.Thus, MobileNetV2 is lightweight because of depth-separable convolution architecture and inverted residual bottleneck layers.

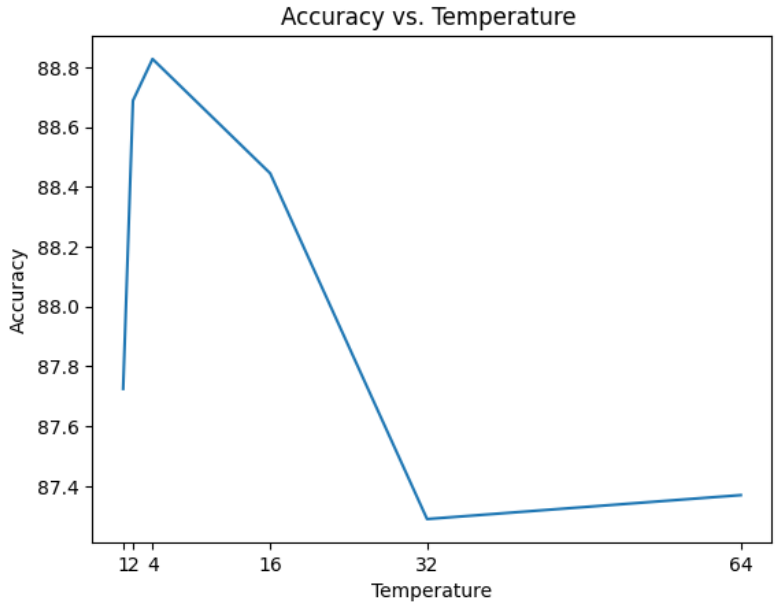
2. 2. 1 Code Implementation Explanation

In Task 2, the dataset was first loaded, with a specific focus on delineating two class labels as our target variable y, while the images were designated as the independent variable x. Subsequent to this, transformation techniques were employed to convert images into tensor format, accompanied by normalization to standardize the dataset, ensuring consistent input feature scaling.

The modeling framework was underpinned by the architectures of esteemed pre-trained models: ResNet50V2 and MobileNetV2. Rather than retraining the entire architecture, layers from these pre-trained models were retained in their learned state, with an appended Fully Connected Neural Network (FCNN) classifier facilitating the fine-tuning phase.

The loss function adopted mirrors that of our preceding task. The teacher loss computes the cross-entropy loss between the true labels and the logits. We implement knowledge distillation for the student model. The combined student loss encompasses both the standard cross-entropy loss and the distillation loss. Initially, the temperature will be setted as 4 and alpha is 0.5. Owing to the test dataset's imbalanced nature, the Area Under the Curve (AUC) metric was chosen as a paramount indicator of model performance. The training regimen was structured such that for the initial 10 epochs, only the feature extraction layers were refined, achieved by fixing the trainability status of the pre-existing layers. This was followed by a 25-epoch phase wherein all model layers were rendered trainable.

Evaluation pivoted on computing the AUC for each batch in the test set, culminating in an aggregated average AUC as the ultimate performance metric. Distinct loss functions were established for the teacher and student models. Notably, the student model developed from scratch eschewed Knowledge Distillation, warranting the adoption of a standard cross-entropy function for training.



2.2.2 FLOPs and XAI Methodologies Implement

Comparing the teacher and student model: We calculate the FLOPs (Floating Point Operations Per Second) for the teacher model, student model with KD, and student model without KD by using the get_flops function. Computing FLOPs is valuable when assessing the computational efficiency and resource usage of a neural network model. The get_flops function takes a TensorFlow model and outputs the FLOPs value. Meanwhile, there is a wrapper function to define the model forward pass. From the table, the Flops of the teacher model is 223296578112 with batch number 32, and the FLOPs for two student models is 19258804800. Meanwhile, we print out the summary for those three models. The total params for the teacher model is 25615802 (97.72 MB) and the total params for two student models is 3540986 (13.51 MB). Thus, the student model is lighter than the teacher model and with smaller FLOPs.

FLOPs of Teacher Model:	223296578112
FLOPs of Student Model With KD:	19258804800
FLOPs of Student Model Without KD:	19258804800

LIME is a method in the realm of Explainable AI (XAI) designed to elucidate the predictions of machine learning classifiers. The sample image will be collected from the first batch of train_loader. In order to get the explanation with the LIME method, we reference the code from Mohammadi and Seyed Mahmoud [10]. There are three explanation maps, from teacher model, student model with KD and student model without KD. The Figure 3.12 shows the partial results of the XAI method output.

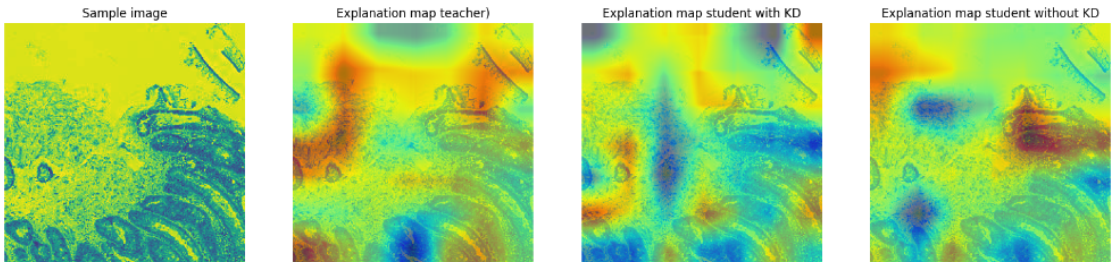


Figure 3.12: Partial Results Of The LIME methods.

In Figure 3.12, there are 4 images in total and they represent an original image and the 3 explanations with the teacher model, student model with KD and the student model without KD. In Figure 3.12, the orange-red zones indicate regions with the highest importance, while the blue zones denote less influential areas. By comparing the explanation map from the teacher model and the student model, there are some features in the sample image that will be focused by both the teacher model and the student model. By comparing the explanation map from the teacher model and the student model, the orange-red zones appeared in different areas, which means the highest importance is different for those two models. However, the explanation map of the student model with KD has some similar orange-red zones and blue zones. Thus, we can conclude that the explanation map of the student model with KD is more aligned with the teacher's, implying that it has "learned" some of the teacher's focus points.

2.2.a Implementation of Proposed Method of Paper No.15

Incorporating insights from the referenced research paper, Subclass Knowledge Distillation (SKD) was implemented. The dataset revealed a noteworthy column, 'Number of Annotators who Selected SSA (Out of 7)', prompting the integration of additional subclasses. Consequently, labels were segmented into four nuanced categories: 'Strongly Disagree', 'Disagree', 'Agree', and 'Strongly Agree' with SSA.

A pivotal modification entailed a unique computation approach for the student loss. The rationale was grounded in the belief that varying hamming distances in one-hot label representations should influence the loss magnitude. Specifically, smaller distances, indicative of closely related classes, would yield a relatively smaller loss, whereas pronounced differences would escalate the loss. This philosophy birthed the introduction of a 1.5x weighting factor to the loss for pronounced differences. The remaining training trajectory mirrored previous methodologies. The final AUC for the student model with SKD is 82.17.

Model	AUC
Student with KD	80.09
Student with SKD	82.17

2.2.b Implementation of Proposed Method of Paper No.9

We implement the Teacher Assistants Knowledge Distillation (TAKD) framework. The data will be set by the following: Temperature = 4, ALPHA = 0.5, and Initial Epoch = 10, Fine-Tuning Epoch =

25, Learning Rate for teacher is 1e-3 and the Learning rate for TA and student is 1e-4. The TA model is the bridge between the teacher model and the student model so we choose a model with the total number of params between the teacher model (25618805 (97.73 MB)) and the student model (121 3543989 (13.52 MB)), DenseNet (8067509 (30.78 MB)). Additionally, there is an improvement in task 2 TA implementation comparte to the task 1 is that load the Data part, the data will be separated into 5 classes: HP, SSA (0 <= annotator count <= 4), SSA (annotator count == 5), SSA (annotator count == 6) and SSA (annotator count == 7). The accuracy for the student model with TAKD framework is 79.29% and the accuracy for the BLKD is 76.24% and the accuracy for the teacher model is 80.84%, accuracy for the TA model is 80.49%. Thus, TAKD works to increase the accuracy of the student model. However, the accuracy of the student model with 2 classes is 80.09%, and the new teacher model and the student model have lower accuracy than it. In the future it is an idea to test the different parameters.

2.3 Explaination

In our experimental setup, both the teacher and student models were constructed without leveraging the weights pre-trained on the "ImageNet" dataset. Specifically, this was achieved by setting the weights parameter to None within our implementation.

When evaluating the model performance using the Area Under the Curve (AUC) metric, maintaining consistent hyperparameters, the teacher model constructed upon the ResNet50V2 architecture yielded an AUC of 85.13. Concurrently, the student model built on the MobileNetV2 architecture, enhanced with Knowledge Distillation (KD), reported an AUC of 80.09. However, upon excluding the utilization of pre-trained weights, there was a notable decline in performance. The AUC plummeted to 68.50 for the teacher model and 62.20 for the student model.

Models pre-trained on extensive datasets, such as ImageNet, benefit from the nuanced and diverse feature representations learned from millions of images. The importance of this is accentuated in our scenario where the datasets used for training and testing are imbalanced, posing challenges for generalization. With a limited dataset, the value of a student model lies in its capacity for feature extraction and fine-tuning. Yet, utilizing this same dataset for both feature extraction and classification can be even more advantageous. As previously discussed, transfer learning serves as a potent regularization technique. Beginning with already generalized features, the model is better equipped to circumvent overfitting.

Transitioning to the aspect of knowledge transfer, a direct comparison between the KD-enhanced student model and its counterpart trained from scratch provides illuminating insights. With consistent hyperparameters, the KD-augmented student model boasted an AUC of 80.09, while its counterpart registered an AUC of 73.08. This underscores the efficacy of knowledge distillation in transposing the intricate knowledge embedded within a sophisticated teacher model into a streamlined student model. The student, thus, can approximate the performance of the teacher model more closely than anticipated. The dual advantages bestowed upon the student model—ingesting knowledge from the teacher and gleaning features from potential pre-trained weights—can dramatically uplift its performance, especially juxtaposed against a student model trained in isolation.

To encapsulate, transfer learning and knowledge distillation emerge as formidable methodologies to harness pre-existing knowledge. Whether sourced from a model pre-trained on a vast dataset or distilled from a more complex teacher model, these techniques can substantially amplify the performance of a student model.

Model	AUC
Teacher with pre-trained	85.13
Teacher without pre-trained	68.86
Student with pre-trained with KD	80.09
Student with pre-trained from scratch	73.08
Student with pre-trained with SKD	82.17
Student without pre-trained	62.19

Figure 3: difference Model with different tech's AUC comparison

Implement Code

GitHub: [ECE1512-2023F-ProjectRepo-Lanzhi-Xiao-Bocheng-Zhang/ProjectA at main · zbc0917/ECE1512-2023F-ProjectRepo-Lanzhi-Xiao-Bocheng-Zhang (github.com)]

References:

[1] C. Buciluˇa, R. Caruana, and A. Niculescu-Mizil. Model compression. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’06, pages 535–541, New York, NY, USA, 2006. ACM.

[2] [1] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," in NIPS 2014 Deep Learning Workshop, arXiv:1503.02531, 2015. [Online]. Available: https://arxiv.org/abs/1503.02531

[3] M. Tzelepi, N. Passalis, and A. Tefas, “Efficient online subclass knowledge distillation for image classification.” in Proceedings of International Conference on Pattern Recognition (ICPR), 2021, pp. 1007–1014.

[4] S. I. Mirzadeh, M. Farajtabar, A. Li, N. Levine, A. Matsukawa, and H. Ghasemzadeh, “Improved knowledge distillation via teacher assistant”, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 5191–5198. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/5963/581

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Identity Mappings in Deep Residual Network," Microsoft Research, 2016. Available: arXiv:1603.05027v3.

[6] M. Sandler, A. Howard, M. Zhu, A. Zhigunov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," Google Inc, 2019. [Online]. Available: arXiv:1801.04381v4.

[7] A. Sajedi, Y. A. Lawryshyn and K. N. Plataniotis, "Subclass Knowledge Distillation with Known Subclass Labels," 2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP), Nafplio, Greece, 2022, pp. 1-5, doi: 10.1109/IVMSP54334.2022.9816232.

[8]A. Brital, "Residual Networks With Examples," Medium, [Online]. Available: https://medium.com/@AnasBrital98/residual-networks-with-examples-80b47cacecf4.

[9]Tokusumi, "flops_calculation.py," keras-flops, 2020. [Online]. Available: https://github.com/tokusumi/keras-flops/blob/master/keras_flops/flops_calculation.py.

[10]Mohammadi and S. Mahmoud, "xai_utils.py," ECE1512_2022W_ProjectRepo, GitHub, 2023. Available:https://github.com/RezaMohammadi99/ECE1512_2022W_ProjectRepo_Seyedmahmoud-Mohammadi/blob/main/Project_A/xai_utils.py.