探讨知识蒸馏及可视化对边缘部署的影响

孙屹 姚昊普

北京商汤科技开发有限公司，北京市海淀区，100089；

摘要：深度网络已经构建了深度架构，表现出令人信服的准确性和良好的收敛行为。确实，增加网络深度可以提高性能，但这也会导致基于梯度的训练变得困难的问题，因为更深的网络往往更加非线性。在如此深的神经元网络中泛化预测的计算成本可能太高,无法让大量人员在现实生活中进行部署。为了便于部署，需要实施知识压缩技术，将知识从繁琐的网络压缩成简化模型。在本报告中，将介绍知识蒸馏并将其应用于 MNIST 和 MHIST 数据集。相比之下,将实施早期停止方法,以看到与传统知识蒸馏相比的改进。在 MHNIST 的实验中，将应用迁移学习技术来了解具有微调泛化预测的预训练模型如何使用预训练的 DOF 进行微调。

关键词：知识提炼；迁移学习；提前停止；KD；ESKD；XAI

DOI：10.64216/3080-1508.25.02.056

1 简介

深度学习技术已广泛应用于人工智能，包括面部识别等图像处理中的不同应用。但是，现实生活中的系统通常涉及一个主要网络，该网络由具有数百万个参数的极深和极宽的网络组成。训练这样的系统非常耗时,因为调整这些参数需要进行大量的计算。此外，为了实现繁琐的系统，对巨大的内存空间也有很高的要求。因此，广泛部署如此繁琐的网络是不可能实现的。

处理这些限制的最常见方法之一是模型压缩。简化模型通过压缩原始模型来减小大小或延迟。但是，这导致了另一个问题，即简化模型应该在不显著降低准确性的情况下实现目标。在本报告中，将介绍知识蒸馏和迁移学习技术，以保证在实现压缩模型以处理冗余数据集时的准确性需求。

2 蒸馏：转移类型概率

2.1 为什么使用知识蒸馏

在现实生活中创建和部署的模型通常具有从高度冗余的数据集中训练提取的复杂结构。要观察模型性能，训练模型是必不可少的，但它可能会花费大量的训练时间。许多笨重的模型需要几周甚至几个月的时间才能完成完全训练。相反，与繁琐的模型相比，训练参数较少的轻量级模型所需的时间更少,但它可能无法处理复杂的目标。一旦训练了繁琐的模型，更适合部署的学生模型就可以作为教师模型,将知识转移到更适合部署的学生模型。

2.2 什么是知识转移

实施深度网络的目标是很好地推广到新数据。训练过程对于模型很好地泛化至关重要,但这种对所需目标信息的需 求允许模型正确泛化。但是，所需信息通常不可用。通过单独训练两种类型的模型，由不同模型的 lager 集成组成的繁琐模型通常比小模型具有更强的学习能力。这导致实际上教师模型能够获得可能存在的暗知识，并且可能是相关的， 但由于尚未确定所分析主题的边界而尚未被发现。通过应用知识蒸馏，学生模型可以获得那些暗知识，以便在教师模 型的帮助下在泛化方面表现更好。所以，知识提炼技术基本上就是把泛化能力的知识从繁琐的模型转移到小模型上。

2.3 基本理念

在神经元网络中，输出决策取决于类概率，通常通过实现“softmax”输出层来推广。“softmax”输出层将为每个类计算的 logit 转换为概率。此过程基于以下公式：

$$q_i = \frac{(\exp(z_i/T))}{(\sum_j \exp(z_i/T))} \quad (1)$$

其中 xi 是“softmax ”层上每个神经元的对数，qi 是对应于每个类别的概率，T 是用于平滑概率分布的温度。如果模型单独训练，则 T 通常设置为 1，因为模型的泛化结果直接取决于类别概率。在知识蒸馏中，这是完全不同的。教师模型不是简单地转移概率，而是将知识或信息传递给学生模型。概率分布与可用于描述信息量的熵密切相关。在数学中，科学家们已经证明，更高的熵意味着可以描述更多的信息。通过增加 T，概率分布变得更加均匀，从而导致更高的熵。因此，随着熵的增加，在知识蒸馏中，更多的信息可以从教师模型转移到学生模型。

2.4 公式分析

在常规知识蒸馏中，学生网络被训练来优化以下损失函数：[1]

$$L_{KD} = L_{original} + \alpha L_{distillation} \tag{2}$$

损失函数由两部分组成。原始损失是通过比较学生和目标标签的决策输出来计算的交叉熵。如果我们希望学生模型专注于从老师那里学到的知识，则应增加 α 以使蒸馏损失在总损失中占有更高的份额。

蒸馏损失是通过比较教师和学生模型的决策结果得出的。更准确地说，它是通过应用教师和学生的软化 logits 之间的 KL 散度来计算的。蒸馏损失的公式构建如下：

$$\sum_{x_i} KL(softmax(f_t(x_i)/T)), softmax(f_t(x_i)/T)) \tag{3}$$

其中 ft xi 是对应于教师和学生模型的单独类的概率分布。超参数 T 是使概率分布更柔和的温度。KL 代表计算相对熵的 KL 发散作。

LKD 函数的优化过程与交叉熵损失函数非常相似。梯度下降公式可视化如下 [1]：

$$\frac{\partial C}{\partial z_i} = \frac{1}{T}(q_i - p_i) = \frac{1}{T}(\frac{e^{(z_i/T)}}{\sum_j e^{(z_j/T)}} - \frac{e^{(v_i/T)}}{\sum_j e^{(v_j/T)}}) \tag{4}$$

如果我们现在假设每个分动壳的 logit 都单独为零均值。方程 (4) 可以简化为以下 [1]：

$$\frac{\partial C}{\partial z_i} = \frac{1}{NT^2}(z_i - v_i) \tag{5}$$

2.5 更多细节

知识蒸馏具有与正则化误差类似的功能。通过传递知识，学生从教师模型中学习预测的趋势，从而提高泛化的准确性。因此，它也可以被视为一种正则化技术。

3 知识蒸馏：提前停止

本文的主要思想是作者构建了教师模型的不同变体，然后使用它们将知识提炼到学生模型中。根据结果，经过训练的教师模型具有高精度并不意味着更好的蒸馏。但是，与完美教师模型相比，一些不完美的教师模型对具有相同超参数的学生模型有更好的贡献。

在论文中，作者发现，与将教师模型训练到完美状态相比，实施非完美教师有时会产生更好的学生模型。造成这种情况的原因是，完美的教师模型很难让学生学习知识的提炼，因为学生没有足够的 DOF 来赶上教师模型的行为。因此，正常知识蒸馏与该方法的区别在于教师模型训练的纪元。在本文中，作者使用了教师模型，而没有进行足够的蒸馏训练。并显示了与普通知识蒸馏学生模型相比更好的学生模型。

这种方法的局限性在于我们如何找到训练教师模型的 epoch 数。如果纪元数太低，教师模型可能会表现不佳，并将错误的知识提炼到学生模型中。另一方面，如果纪元数太大，教师模型将具有完美的性能和准确性，但学生模型将很难赶上学习。本文没有过多地讨论这个限制。单独来说，我认为这个问题可以通过多次运行教师模型的不同数字 epoch 然后找到最佳 epoch 数字数来解决。

4 迁移学习

迁移学习是提高机器学习中训练过程性能和效率的另一种技术。并不总是为模型提供与特征空间匹配的训练数据来学习兴趣。通过从相关域传输信息来改进一个域的学习者是核心思想。它允许模型在有限的目标训练数据供应下进行训练 [2]。更具体地说，通用模型可以从在足够大且通用的数据集上训练的预训练模型继承技能或特征空间的可视化。

然而，与任何技术一样，使用机器学习进行迁移学习也有其自身的挑战。在这一点上，负转移问题是最大的限制之一。如果初始问题和目标问题足够相似，以至于第一轮训练具有相关性，则迁移学习有效。迁移学习的更多细节将在任务 2 中根据训练 MHIST 的表现进行讨论

5 MNIST

在任务 1 MNIST 部分，我们有数字 0 到 9 的手写图像。我们想构建两个模型。教师模型和学生模型。然后利用知识蒸馏，将黑暗知识从教师模式提炼到学生模式。

5.1 模型设计

在本例中，我们创建两个模型。一种是教师模型，它更复杂、更准确，另一种是学生模型，它需要更少的计算，在实　　际中运行得更快。图 (1) 显示了教师模型的详细信息。图 (2) 显示了学生模型的细节。我们可以看到，教师模型的参数

1625866，学生模型的 1238730 参数相对比教师模型少得多。

```
Model: "teacherNet"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 28, 28, 32)        320

 max_pooling2d (MaxPooling2D  (None, 28, 28, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 28, 28, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 flatten (Flatten)           (None, 12544)             0

 dropout (Dropout)           (None, 12544)             0

 dense (Dense)               (None, 128)               1605760

 dropout_1 (Dropout)         (None, 128)               0

 logits (Dense)              (None, 10)                1290

 softmax (Activation)        (None, 10)                0

=================================================================
Total params: 1,625,866
Trainable params: 1,625,866
Non-trainable params: 0
_________________________________________________________________
```

Fig. 1. Teacher model (CNN model)

```
Model: "studentNet"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 784)               0

 dense (Dense)               (None, 784)               615440

 dense_1 (Dense)             (None, 784)               615440

 logits (Dense)              (None, 10)                7850

 softmax (Activation)        (None, 10)                0

=================================================================
Total params: 1,238,730
Trainable params: 1,238,730
Non-trainable params: 0
_________________________________________________________________
```

Fig. 2. Student model (FC model)

5.2 模型损失函数

详细信息在代码文件中。在这一部分中，我们定义了由教师模型输出和 true 标签计算的教师损失函数。学生来源损失也以类似的方式定义。对于蒸馏部分，它需要 student logits 输出、teacher logits 输出和超参数 T。蒸馏成本是通过首先计算教师和学生的软 logits 来计算的。然后计算教师软 Logits 和学生软 Logits 之间的 KL 散度。最后，学生总损失是通过将原点损失和蒸馏损失乘以 α 相加来计算的，其中 α 是另一个超参数。

5.3 训练和评估功能

详细信息在代码文件中。在这部分，我们定义了 train 和 evaluate 函数。图 (3) 显示了学生模型（FC 模型）的准确率，T = 4。

```
Epoch 1: Class_accuracy: 98.00%
Epoch 2: Class_accuracy: 98.30%
Epoch 3: Class_accuracy: 98.42%
Epoch 4: Class_accuracy: 98.57%
Epoch 5: Class_accuracy: 98.70%
Epoch 6: Class_accuracy: 98.68%
Epoch 7: Class_accuracy: 98.58%
Epoch 8: Class_accuracy: 98.67%
Epoch 9: Class_accuracy: 98.67%
Epoch 10: Class_accuracy: 98.68%
Epoch 11: Class_accuracy: 98.57%
Epoch 12: Class_accuracy: 98.71%
98.71
```

Fig. 3. Student model training accuracy

5.4 温度的模型精度

图 (4) 显示了教师模型的准确性。最后，准确率约为 98% 到 99%。对于学生模型精度，如图 (5) 和 (8) 所示。我们可以看到，在 MNIST 数据集中，温度对学生模型的准确性影响不大。原因可能是我们的教师模型准确率在 98% 到 99% 左右，所以我们的学生模型相对接近我们的教师模型准确率。因此，温度变化不会对学生模型的准确性产生太大影响。然而，从精度与温度的图 (9) 中，我们可以发现即使是精度也没有太大的差异。它仍然随着温度的升高而增加。这意味着 teacher 的更统一的排列会将更多信息传递给 student。如果学生能够处理这些信息。然后准确性会提高。

```
Epoch 1/12
234/234 [==============================] - 12s 12ms/step - loss: 0.2892 - accuracy: 0.9123
Epoch 2/12
234/234 [==============================] - 3s 11ms/step - loss: 0.1000 - accuracy: 0.9701
Epoch 3/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0762 - accuracy: 0.9773
Epoch 4/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0641 - accuracy: 0.9809
Epoch 5/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0564 - accuracy: 0.9826
Epoch 6/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0508 - accuracy: 0.9846
Epoch 7/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0443 - accuracy: 0.9862
Epoch 8/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0422 - accuracy: 0.9865
Epoch 9/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0370 - accuracy: 0.9883
Epoch 10/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0369 - accuracy: 0.9887
Epoch 11/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0330 - accuracy: 0.9895
Epoch 12/12
234/234 [==============================] - 3s 11ms/step - loss: 0.0306 - accuracy: 0.9902
```

Fig. 4. Teacher model training and testing accuracy

```
Epoch 1: Class_accuracy: 96.88%
Epoch 2: Class_accuracy: 97.49%
Epoch 3: Class_accuracy: 97.89%
Epoch 4: Class_accuracy: 97.98%
Epoch 5: Class_accuracy: 97.73%
Epoch 6: Class_accuracy: 98.52%
Epoch 7: Class_accuracy: 98.15%
Epoch 8: Class_accuracy: 97.89%
Epoch 9: Class_accuracy: 98.35%
Epoch 10: Class_accuracy: 98.41%
Epoch 11: Class_accuracy: 98.06%
Epoch 12: Class_accuracy: 98.16%
In temperature = 1.0, accuracy after 12 epoch is: 98.159996.
```

Fig. 5. Student model training and testing accuracy

T = 1

```
Epoch 1: Class_accuracy: 96.51%
Epoch 2: Class_accuracy: 97.77%
Epoch 3: Class_accuracy: 98.06%
Epoch 4: Class_accuracy: 98.39%
Epoch 5: Class_accuracy: 98.51%
Epoch 6: Class_accuracy: 98.52%
Epoch 7: Class_accuracy: 98.65%
Epoch 8: Class_accuracy: 98.66%
Epoch 9: Class_accuracy: 98.69%
Epoch 10: Class_accuracy: 98.81%
Epoch 11: Class_accuracy: 98.66%
Epoch 12: Class_accuracy: 98.71%
In temperature = 4.0, accuracy after 12 epoch is: 98.71.
```

Fig. 6. Student model training and testing accuracy

T = 4

```
Epoch 1: Class_accuracy: 96.77%
Epoch 2: Class_accuracy: 98.12%
Epoch 3: Class_accuracy: 98.31%
Epoch 4: Class_accuracy: 98.50%
Epoch 5: Class_accuracy: 98.59%
Epoch 6: Class_accuracy: 98.69%
Epoch 7: Class_accuracy: 98.67%
Epoch 8: Class_accuracy: 98.70%
Epoch 9: Class_accuracy: 98.77%
Epoch 10: Class_accuracy: 98.75%
Epoch 11: Class_accuracy: 98.79%
Epoch 12: Class_accuracy: 98.83%
In temperature = 16.0, accuracy after 12 epoch is: 98.83.
```

Fig. 7. Student model training and testing accuracy

T = 16

```
Epoch 1: Class_accuracy: 96.85%
Epoch 2: Class_accuracy: 98.00%
Epoch 3: Class_accuracy: 98.26%
Epoch 4: Class_accuracy: 98.53%
Epoch 5: Class_accuracy: 98.70%
Epoch 6: Class_accuracy: 98.68%
Epoch 7: Class_accuracy: 98.74%
Epoch 8: Class_accuracy: 98.78%
Epoch 9: Class_accuracy: 98.88%
Epoch 10: Class_accuracy: 98.81%
Epoch 11: Class_accuracy: 98.90%
Epoch 12: Class_accuracy: 98.92%
In temperature = 64.0, accuracy after 12 epoch is: 98.92.
```

Fig. 8. Student model training and testing accuracy

T = 64

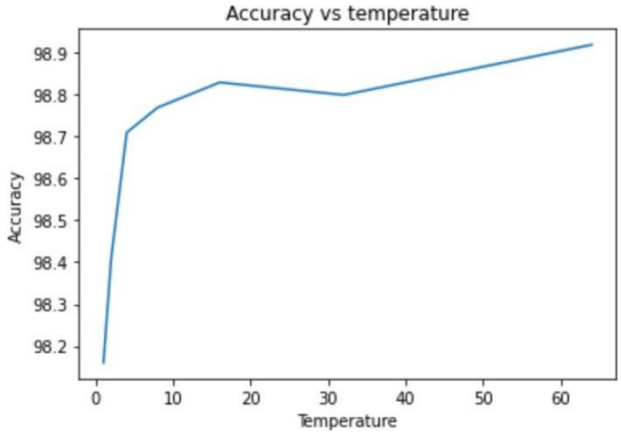

Fig. 9. Student model Accuracy vs Temperature

6 无蒸馏学生模型精度

如图 (10) 所示，即使没有知识蒸馏损失。我们的学生模型仍然能够达到 98% 的准确率。这意味着我们的学生模型（FC 模型）足够复杂，可以处理 MNIST 数据集。并且它还显示了精度与温度部分的原因，学生模型的准确性对蒸馏温度不敏感。然而，与图 (9) 相比，我们可以发现 98.15% 与温度 = 1 学生模型准确率的 KD 相似。这也意味着在 MNIST 数据集上，即使提供额外的信息也很难改进学生模型。在这种情况下，更多信息通常会产生更好的模型。

```
Epoch 1: Class_accuracy: 96.10%
Epoch 2: Class_accuracy: 97.64%
Epoch 3: Class_accuracy: 97.78%
Epoch 4: Class_accuracy: 97.84%
Epoch 5: Class_accuracy: 98.02%
Epoch 6: Class_accuracy: 97.75%
Epoch 7: Class_accuracy: 98.06%
Epoch 8: Class_accuracy: 98.11%
Epoch 9: Class_accuracy: 98.05%
Epoch 10: Class_accuracy: 97.87%
Epoch 11: Class_accuracy: 98.10%
Epoch 12: Class_accuracy: 98.21%
98.21
```

Fig. 10. Student model without Knowledge Distillation loss

6.1 FLOPs

如图 figure(11) 所示，教师模型的 FLOPs 几乎是学生模型的 10 倍。KD 学生模型和非 KD 学生模型具有相同的权重。这会导致它们具有相同的 FLOP。从这部分我们可以发现 CNN 层需要大量的计算。因为教师的参数只比学生模型多 33%，比学生模型多 10 倍。教师和学生之间的区别在于教师有几个 CNN 层，而学生只有密集层。

```
For teacher model:
FLOPS: 0.0328G
For student model:
FLOPS: 0.00248G
For student model without distillation:
FLOPS: 0.00248G
```

Fig. 11. FLOPs for teacher and student model

6.2 XAI

在这个项目中，我们需要使用一种同时兼容教师模型和学生模型的 XAI 方法。某些方法（如 Grad-CAM 或 Grad- CAM++）需要 CNN 层，将与学生模型不兼容。因此，我们选择 RISE 方法进行模型可视化。如图 (12) (13) (14) (15) 所示，教师模型和具有

知识蒸馏的学生模型具有相似的关注区域。而没有知识提炼的学生模式将有不同的关注领域。

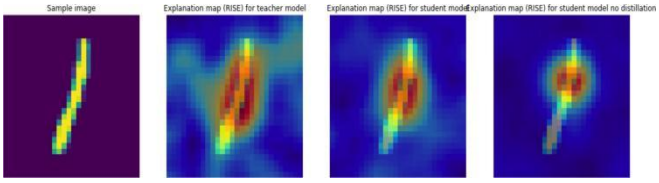

Fig. 12. RISE for model visualization number 1

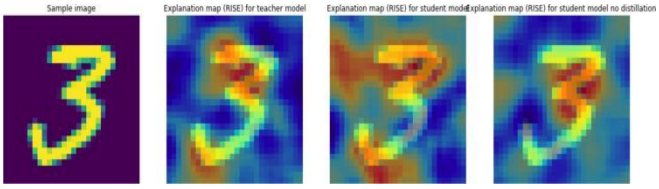

Fig. 13. RISE for model visualization number 3

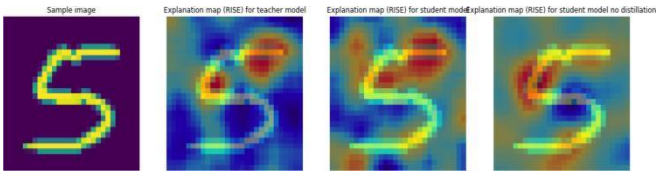

Fig. 14. RISE for model visualization number 5

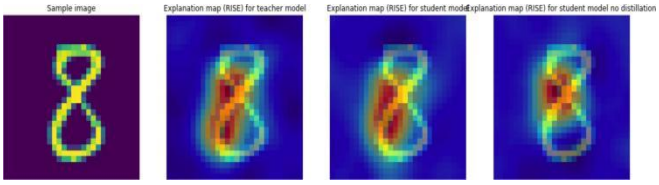

Fig. 15. RISE for model visualization number 8

6.3 提前停止知识蒸馏

在这部分，我们选择论文 [2] 方法。在本文中，它主要讨论一种称为提前停止的方法。这种方法的主要思想是在教师模型变得完美之前停止训练，学生模型将更容易学习到知识被教师模型提炼出来。

在这部分，我用 1 到 25 的纪元模型训练教师模型，并使用这些教师模型将知识提炼到学生模型中。从图 (16) 中，x 轴是 epoch 数，y 轴是 accuracy。我们可以看到 epoch 只对教师模型产生相对较大的影响。每个 epoch 编号期间的学生模型准确性几乎没有变化。这意味着在 MNIST 数据集上，提前停止方法对知识蒸馏没有太大的优势。然后原因是早停被用来解决教师模型太复杂而无法被学生模型学习的问题。但在上一节中。我们发现，即使没有蒸馏的学生模型仍然能够找到这个数据集的解。因此，从教师模型中提炼出的知识对于学生模型来说就不需要捕捉以提高准确性。这导致学生模型对不同的时代教师模型不敏感。

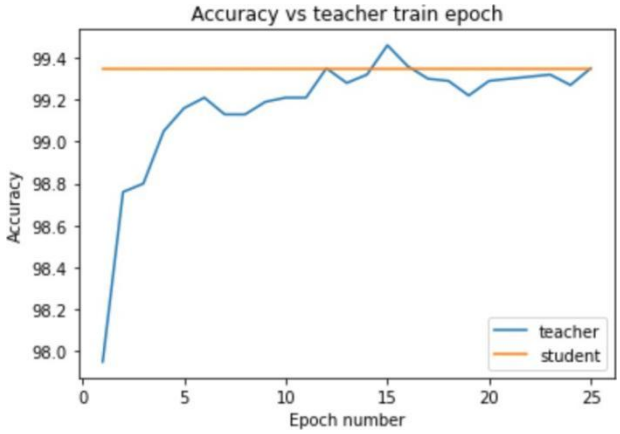

Fig. 16. Model Accuracy vs Epoch number

7 MHIST

MHIST 数据集包含人体组织图像，标签分为 HP 和 SSA 两类。本研究结合迁移学习与知识蒸馏方法，训练轻量级模型 MobileNetV2 模拟教师模型 ResNet50V2 的表现。

7.1 模型架构与设置

在模型结构方面，我们加载预训练的 ResNet50V2 和 MobileNetV2，固定除最后一层以外的所有参数，仅训练 logits

层，并在其上添加 Flatten、Dense 和 Activation 层，以适应二分类任务。

•ResNet50V2 相较于 ResNetV1 引入了归一化与激活顺序的调整，提升了在 ImageNet 等数据集上的表现。

•MobileNetV2 使用倒置残差结构与深度可分离卷积，大幅减少模型参数和计算复杂度。

此外，ResNet 架构通过引入残差连接，有效缓解了梯度消失问题，使得深层网络仍能获得稳定训练效果。

7.2 训练与蒸馏过程

我们采用如下训练策略：

•教师模型训练 6 个 epoch，学生模型训练 8 个 epoch；

训练仅涉及最后几层参数；

温度调节的影响

在 Softmax 温度 T 的控制下，我们发现 T = 4 时学生模型的准确率最高。温度过低时，学生无法有效学习教师模型的“暗知识”；温度过高时可能忽略原始标签信息，造成精度下降。这表明 MHIST 数据集的复杂性导致学生模型更依赖于来自教师的额外信息。

无蒸馏对比

在没有知识蒸馏的帮助下，学生模型准确率在 68%-77% 之间波动，表现不稳定，说明蒸馏机制在该任务中并非始终有益，但可在特定条件下增强学生模型的泛化能力。

7.3 计算资源分析

通过 FLOPs 对比我们发现：

•教师模型的计算量是学生模型的约 11 倍；

•KD 学生与非 KD 学生结构相同，FLOPs 相同；

•MobileNetV2 在计算资源受限场景下具有明显优势。

7.4 可解释性（XAI）分析

使用 RISE 方法进行模型可视化：

•教师模型与 KD 学生模型关注区域较为一致；

•无蒸馏学生模型则可能聚焦于无关区域；

•错误预测情况表明教师模型偶尔会将错误信息传递给学生；

•某些图像中三个模型的关注区域完全不同，说明特定输入可能存在多种判别策略。

7.5 早期停止策略

我们尝试不同的教师训练 epoch 数（如 12 与 25）：

•训练轮数过少时教师模型表现不佳，蒸馏效果差；

•训练轮数过多时，教师可能学习出学生无法模拟的复杂解；

•学生模型精度在教师训练 epoch 过高时不稳定，可能被误导。

7.6 迁移学习与知识蒸馏效果总结

通过迁移学习，学生模型继承了预训练模型的大部分特征表示，显著加快了训练速度。而知识蒸馏进一步提升了学生 模型在某些条件下的精度。然而，蒸馏效果强烈依赖于教师模型是否在学生模型表示能力范围内构建了解决方案。若教师模型解过于复杂或偏离学生空间，反而可能对学生模型产生负面影响。因此，在 MHIST 数据集上，蒸馏效果依赖于教师模型训练轮数的合理选择。

8 结论

我们已经证明，知识蒸馏通常通过将知识从高度正则化模型转移到蒸馏模型，对提高泛化性能具有积极影响。在 MNIST 上，蒸馏会产生良好的性能，学生几乎具有与教师模型相同的预测准确性。然而，蒸馏也有局限性。如果繁琐的输出精度低于一定水平，则无法起到老师的作用。学生会从老师那里学到不正确的知识，与学习本身相比，情况可能更糟。因此，它不会显著提高 MHIST 的性能。一般来说，知识蒸馏是一种可靠的技术，它使用户更容易部署深度网络。

参考文献

[1]G.Hinton,O.Vinyals,and J.Dean,“Distilling the Knowledge in a Neural Network,”arXiv e-prints,p.arXiv:1503.02531,Mar.2015.

[2]J.H.Cho and B.Hariharan,“On the Efficacy of Knowledge Distillation,”arXiv e-prints,p.arXiv:1910.01348,Oct.2019.