Knowledge Distillation with Distribution Mismatch

Dang Nguyen ✉, Sunil Gupta, Trong Nguyen, Santu Rana, Phuoc Nguyen, Truyen Tran, Ky Le, Shannon Ryan, and Svetha Venkatesh

Applied Artificial Intelligence Institute (A^2I^2), Deakin University, Geelong, Australia
{d.nguyen, sunil.gupta, trong.nguyen, santu.rana, phuoc.nguyen, truyen.tran, k.le, shannon.ryan, svetha.venkatesh}@deakin.edu.au

Abstract. Knowledge distillation (KD) is one of the most efficient methods to compress a large deep neural network (called *teacher*) to a smaller network (called *student*). Current state-of-the-art KD methods assume that the distributions of training data of teacher and student are identical to maintain the student's accuracy close to the teacher's accuracy. However, this strong assumption is not met in many real-world applications where the distribution mismatch happens between teacher's training data and student's training data. As a result, existing KD methods often fail in this case. To overcome this problem, we propose a novel method for KD process, which is still effective when the distribution mismatch happens. We first learn a distribution based on student's training data, from which we can sample images well-classified by the teacher. By doing this, we can discover the data space where the teacher has good knowledge to transfer to the student. We then propose a new loss function to train the student network, which achieves better accuracy than the standard KD loss function. We conduct extensive experiments to demonstrate that our method works well for KD tasks with or without distribution mismatch. To the best of our knowledge, our method is the first method addressing the challenge of distribution mismatch when performing KD process.

Keywords: Knowledge distillation · Model compression · Distribution mismatch · Distribution shift · Mismatched teacher

1 Introduction

Recently, deep learning has become one of the most successful machine learning techniques [16], and it has been applied widely to many real-world applications including face recognition [8], security systems [21], disease detection [18], recommended systems [25], etc. Deep neural networks (the main component of deep learning) often have millions of parameters (aka weights) to train, thus require heavy computation and storage, which can only be executed on powerful servers. This characteristic renders deep networks inapplicable to many real-time devices, especially for those edge devices with limited resources such as smart phones, autonomous cars, and micro robots.

A well-known solution in machine learning to compress a large deep network to a smaller network is *knowledge distillation* (KD) [9,7]. The main goal of KD is to transfer the knowledge learned by a large pre-trained deep network (called *teacher*) to a smaller network (called *student*) such that the student network can mimic the teacher network, resulting in a comparable classification performance [9]. Many methods have been proposed for KD, and most of them follow the method introduced in Hinton et al.'s paper [9], which attempts to map the predictions of student to both the true labels and the predictions of teacher on the student's training data (called *student-data*). The intuition behind this method is that the student will improve its classification performance when it not only learns from its training data but also is guided by a powerful teacher that was often trained on a larger data (called *teacher-data*) and achieved very good performance due to its generalization ability. The idea of standard KD method is illustrated in Figure 1.

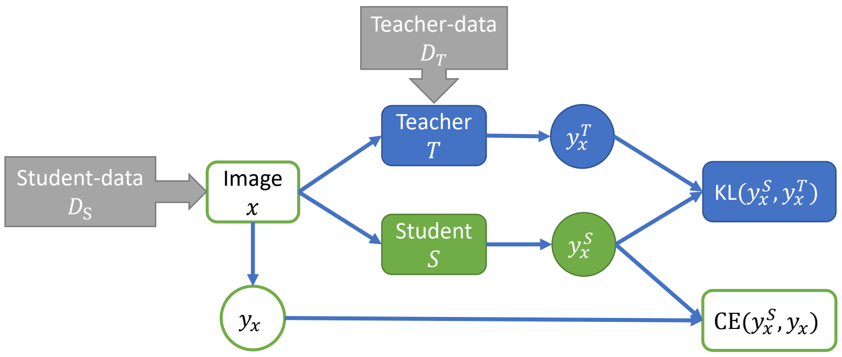

Fig. 1: Illustration of standard KD method. Given a student-data D_S and a teacher network T pre-trained on teacher-data D_T, the student network S is trained on each image $x \in D_S$ such that its output y_x^S matches both the true label y_x via a cross-entropy loss and the output of teacher y_x^T via a Kullback–Leibler (KL) divergence loss.

All current methods following the standard KD method shown in Figure 1 have a significant constraint – they assume that both teacher-data D_T and student-data D_S come from the same distribution [9,10,2,22]. This strong assumption is not realistic in real-world applications, where the distribution shift often happens between D_T and D_S, and in some cases D_T is different from D_S e.g. teacher pre-trained on ImageNet while student trained on CIFAR-100. As a result, the teacher network performs very poorly on D_S. All existing KD methods will fail when using the knowledge transferred from such mismatched teachers, and the classification accuracy of student network often drops significantly. Thus, *the ability of applying the KD process when the distribution mismatch between teacher-data D_T and student-data D_S exists, is an open problem.*

Our method. To solve the above problem, we propose a novel KD method that is still effective when the distribution mismatch happens. In particular, we first train a generative model to obtain the distribution of a latent variable representing the student-data. We then adjust this distribution using Bayesian optimization [19,14] to find an optimized distribution from which we can sample images that are well-classified by the teacher. By doing this, we can replace the *original student-data* where the teacher performs poorly by a *new student-data* where the teacher achieves a good performance. Finally, we propose a novel KD loss function to train the student network to match its predictions to the true labels of *original student-data* and to the predictions of teacher network on *new student-data*. The intuition behind our method is that the teacher network should be given the data points on which it has good knowledge and achieves accurate predictions. By that way, the teacher's good knowledge on such data points will be useful when transferred to the student. For other data points on which the teacher has little/wrong knowledge, the student should learn their information from its own ground-truth labels. Our KD method is robust as it can be applied to two settings of KD process: (1) *with distribution mismatch* and (2) *without distribution mismatch*.

To summarize, we make the following contributions:

1. We propose **KDDM** (*Knowledge Distillation with Distribution Mismatch*), a novel method for distilling the knowledge of a large pre-trained teacher network into a smaller student network. To the best of our knowledge, **KDDM** is the first method offering a successful KD process even if the distributions of teacher-data and student-data are different.
2. We develop an efficient framework to generate images well-classified by the teacher network and a new loss function to train the student network. Both are very useful for the knowledge transfer.
3. We only treat the teacher as a *black-box* model where we require no internal information (e.g. model architecture, weights, etc) from the teacher except its probabilistic outputs.
4. We demonstrate the benefits of **KDDM** in two cases of KD process, namely *with distribution mismatch* and *without distribution mismatch*. Our method significantly outperforms the standard KD method and is comparable with recent state-of-the-art KD methods that train teacher and student networks on data with matching distributions.

2 Related Works

Knowledge distillation (KD) has become an attractive research topic since 2015 when Hinton et al. introduced the concept of KD in their teacher-student framework [9]. The main goal of KD is to transfer the knowledge learned from a teacher network to a student network such that the student can mimic the teacher, resulting in an improvement in its classification performance. In recent years, many methods have been proposed for KD, which can be categorized into three groups: relation-based, feature-based, and response-based KD methods.

Relation-based KD. These methods not only use the output of teacher but also explore the relationships between different layers of teacher when training the student network. Examples include [24,11,15]. One key challenge of these methods is how to define the correlation function between the teacher and student layers.

Feature-based KD. These methods leverage both the output of last layer and the output of intermediate layers (i.e. feature map) of teacher when training student [1,10,15]. The main benefit of these approaches is that deep neural networks are often good at representation learning, therefore not only the predictions but also representations learned by teacher network are useful knowledge to transfer to student network.

Response-based KD. These methods directly mimic the final prediction of teacher network [9,4,12,23]. Compared to relation-based and feature-based methods, response-based methods have a significant advantage that they only treat the teacher as a *black-box* model without requiring access to its internal information (e.g. model parameters, feature maps, or derivatives), thus are applicable to *any* type of deep network.

To successfully train student with the knowledge distilled from teacher, most KD methods assume that both teacher-data and student-data come from the same distribution. For example, [9,3] pointed out that the student only achieved its best accuracy when it had access to the teacher's original training data. Similarly, [13] mentioned the typical setting in existing KD methods is the student network trained on the teacher-data. Recent state-of-the-art methods [10,2,22] also train both teacher and student networks on the same dataset.

Although these methods can distill a large deep network into a smaller one, their success relies on the strong assumption that both teacher-data and student-data are identical, a condition often not met in many real-world applications. As far as we know, no KD approach has been proposed to explicitly address the challenge of distribution mismatch between teacher-data and student-data.

3 Framework

3.1 Problem definition

Given a student-data D_S and a teacher network T pre-trained on a teacher-data D_T, the goal of KD method is to train a student network S on D_S such that S can mimic the prediction of T.

The standard KD method minimizes the following loss function:

$$\mathcal{L}_{KD} = \sum_{x \in D_S} \alpha \mathrm{CE}(y_x^S, y_x) + (1-\alpha)\mathrm{KL}(y_x^S, y_x^T), \tag{1}$$

where $\mathrm{CE}(y_x^S, y_x)$ is the cross-entropy loss between the output (i.e. the probabilities for all classes) of student and true label, $\mathrm{KL}(y_x^S, y_x^T)$ is the Kullback–Leibler (KL) divergence loss between the output of student and the output of teacher, and α is a trade-off factor to balance the two loss terms. Note that in Equation

(1) we do not use the *temperature* factor as in Hinton's KD method [9] because this requires access to the pre-softmax activations of teacher, which violates our assumption of "black-box" teacher.

From Equation (1), since the true labels y_x in the first loss term are fixed, the improvement of a KD method mainly relies on the second loss term $\mathrm{KL}(y_x^S, y_x^T)$. The predictions of teacher y_x^T are assumed to be highly accurate on the student-data D_S so that the classification accuracy of student can be improved. In many real cases, this assumption is not true, where the performance of teacher on D_S is not good due to the distribution mismatch between D_T and D_S. For example, D_T and D_S can come from two different distributions or D_T and D_S can be two different datasets. As a result, the standard KD process makes harmful effects to student, where the classification accuracy of student drops significantly compared to the same model trained from scratch on D_S (we call this model *student-alone*).

Problem statement. Given a student-data D_S and a *black-box* teacher network T pre-trained on a teacher-data D_T, we assume there is a distribution mismatch between D_T and D_S. Our goal is to train a student network S on D_S, which achieves two objectives: (1) our student's classification accuracy is better than that of student network trained with the standard KD loss in Equation (1), and (2) our student's classification accuracy is better than that of student-alone trained from scratch on D_S.

3.2 Proposed method KDDM

To improve the classification performance of student network, one direct solution is to adjust the trade-off factor $\alpha \in [0, 1]$ in Equation (1), where a small value for α means the KD process will rely more on the predictions of teacher whereas a large value for α means the KD process will rely more on the true labels. Typically, existing KD methods choose $\alpha = 0.5$ to balance these two objectives. Since the distribution mismatch exists, a reasonable thinking is that we should trust the true labels more than the predictions of teacher, leading to choosing large values for α (e.g. 0.7 or 0.9). Although this simple approach can improve the accuracy of student, it cannot boost the student's performance better than the accuracy of student-alone network that uses $\alpha = 1.0$. We can see the accuracy of student-alone network serves as an upper bound on the accuracy obtained by the standard KD methods when the distribution mismatch occurs.

Our framework to solve the problem in Section 3.1 is novel, which has three main steps: (1) we train a generative model to obtain the distribution of a latent variable representing the student-data D_S, (2) we adjust this distribution using Bayesian optimization (BO) to find an optimized distribution to generate new images well-classified by the teacher, and (3) we perform the KD process to train the student with a new loss function. The overview of our proposed framework is shown in Figure 2.

Learning the distribution of latent variable z. We train Conditional Variational Autoencoder (CVAE) [20] to learn the distribution of latent variable z

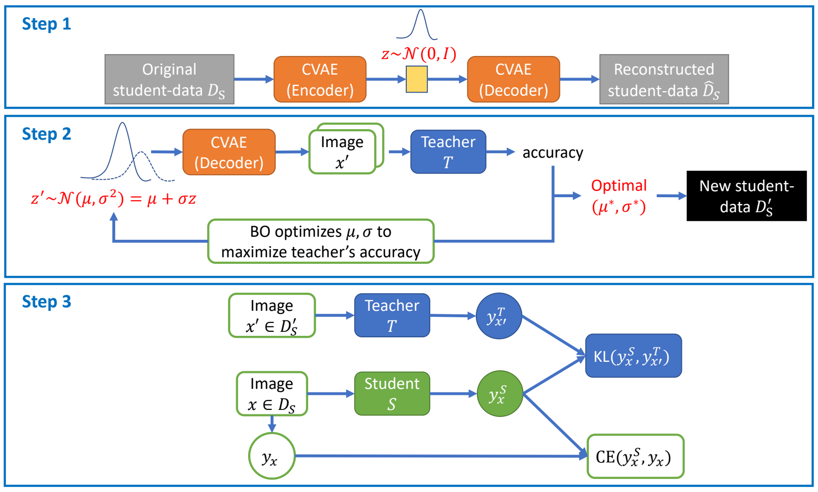
Step 1
Original student-data D_S
CVAE (Encoder)
$z \sim \mathcal{N}(0, I)$
CVAE (Decoder)
Reconstructed student-data $\widehat{D}_S$
Step 2
CVAE (Decoder)
Image x'
Teacher T
accuracy
$z' \sim \mathcal{N}(\mu, \sigma^2) = \mu + \sigma z$
Optimal (μ^*, σ^*)
New student-data D'_S
BO optimizes μ, σ to maximize teacher's accuracy
Step 3
Image $x' \in D'_S$
Teacher T
$y^T_{x'}$
$\mathrm{KL}(y^S_x, y^T_{x'})$
Image $x \in D_S$
Student S
y^S_x
y_x
$\mathrm{CE}(y^S_x, y_x)$

Fig. 2: Our framework **KDDM** consists of three steps. **Step 1:** it trains a CVAE model to learn the standard normal distribution of latent variable z representing the student-data D_S. **Step 2:** it adjusts this distribution using BO to generate the new student-data D'_S where the teacher achieves the best accuracy. **Step 3:** it trains the student to match its output to the true labels in original student-data D_S and to the predictions of teacher on new student-data D'_S.

that is close to the distribution of student-data D_S. CVAE is a generative model consisting of an *encoder* and a *decoder*. We use the encoder network to map an image along with its label $(x, y) \in D_S$ to a latent vector z that follows a standard normal distribution $\mathcal{N}(0, I)$. From the latent vector z conditioned on the label y, we use the decoder network to reconstruct the input image x. Following [20], we train the CVAE by maximizing the variational lower bound objective:

$$\log P(x \mid y) \geq \mathbb{E}(\log P(x \mid z, y)) - \mathrm{KL}(Q(z \mid x, y), P(z \mid y)),$$

where $Q(z \mid x, y)$ is the encoder network mapping input image x and its label y to the latent vector z, $P(x \mid z, y)$ is the decoder network reconstructing input image x from the latent vector z and label y, $\mathbb{E}(\log P(x \mid z, y))$ is the expected likelihood, which is implemented by a cross-entropy loss between input image and reconstructed image, and $P(z \mid y) \equiv \mathcal{N}(0, I)$ is the prior distribution of z conditioned on y.

After training the CVAE model, we obtain the distribution of latent variable z, which is a standard normal distribution $\mathcal{N}(0, I)$. If we sample any $z \sim \mathcal{N}(0, I)$ and feed it along with a label y to the decoder network, then we can generate an image that follows the distribution of D_S. Since we use CVAE to generate images with labels, we can find a distribution to sample images that are well-classified by the teacher in the next step.

Adjusting the distribution of latent variable $\boldsymbol{z}$. We sample latent vectors $z \sim \mathcal{N}(0, I)$; the number of z equals to the number of images in D_S. We then feed these z along with the list of true labels y in D_S to the trained decoder network to generate new images. Since these images are similar to original images in D_S, they will not be predicted well by the teacher, and therefore would not be useful for knowledge transfer.

Our goal is to generate new images well-classified by the teacher. Since z are normally distributed, we adjust the distribution $P(z \mid y) \equiv \mathcal{N}(0, I)$ to $P(z' \mid y) \equiv \mathcal{N}(\mu, \sigma^2)$ using the following formula:

$$z' = \mu + \sigma z, \tag{2}$$

where $z' \in \mathbb{R}^d$ are the new latent vectors, $\mu \in \mathbb{R}^d$ is the mean vector, $\sigma \in \mathbb{R}^{d \times d}$ is the standard deviation matrix that is a diagonal matrix, $z \in \mathbb{R}^d$ are the latent vectors representing the original images in D_S, and d is the dimension of z.

From Equation (2), we can also use the new latent vectors z' along with y to generate images x' via the trained decoder network. The next question is how to generate relevant images x' on which the teacher predicts well. Since the output of generated images x' depends on z' that can be controlled by μ and σ, we use BO to optimize μ and σ by maximizing the following objective function:

$$(\mu^*, \sigma^*) = \arg\max_{\mu, \sigma} f_{acc}(y, T(x')), \tag{3}$$

where $x' = G(z', y)$ are the generated images via the trained decoder network G, $T(x')$ are the predictions of teacher for generated images x', and f_{acc} is the function computing the accuracy between true labels y and predicted labels of teacher on x'. Note that since we set y during the image generation, y are the true labels of x'. We use BO to optimize Equation (3) since it is a black-box and expensive-to-evaluate function.

After the optimization finishes, we obtain the optimal distribution indicated by μ^* and σ^*. We compute optimal latent vectors $z' = \mu^* + \sigma^* z$. Using z' along with the list of true labels y in D_S, we generate images x' whose labels are y, and x' have a higher chance to be correctly predicted by the teacher. On one hand, we can generate a new student-data D'_S containing images x' that are well-classified by the teacher. On the other hand, we can discover the data space where the teacher has relevant knowledge for the student and achieves accurate predictions. Note that for each image $x \in D_S$ we have its corresponding image $x' \in D'_S$, and both x and x' have the same true label y.

Training the student $\boldsymbol{S}$ using a new knowledge distillation loss. After generating the new student-data D'_S, we then use the knowledge of teacher on D'_S to transfer to the student. Precisely, we train a student network whose predictions match both the true labels in the original student-data D_S and the predictions of teacher on the new student-data D'_S.

Our new loss function to train the student network is:

$$\mathcal{L}_S = \sum_{x \in D_S, x' \in D'_S} \beta \mathrm{CE}(y_x^S, y_x) + (1 - \beta) \mathrm{KL}(y_x^S, y_{x'}^T), \tag{4}$$

where $\mathrm{CE}(y^S_x, y_x)$ is the cross-entropy loss between the outputs of student and the true labels in the original student-data D_S, $\mathrm{KL}(y^S_x, y^T_{x'})$ is the KL divergence loss between the outputs of student and the outputs of teacher on the new student-data D'_S, and β is a trade-off factor to balance the two loss terms. Note that the number of images in both D_S and D'_S are the same, and each $x' \in D'_S$ is the coressponding image of each $x \in D_S$.

Compared to the standard KD loss function in Equation (1), our loss function uses the knowledge of teacher on the new student-data D'_S to transfer to the student, which is much better than using the knowledge of teacher on the original student-data D_S. This is because D'_S contains images on which the teacher has good knowledge and predicts accurately, as ensured by the BO process. For the knowledge on D_S where the teacher does not have good knowledge, our student network only learns from the true labels of D_S. Consequently, our method is much better than the standard KD method.

Discussion. To create the new student-data D'_S, one can simply select images in the original student-data D_S on which the teacher has accurate predictions (i.e. the actual classes having the highest probabilities). However, this naive approach has two disadvantages. First, since the teacher's accuracy on D_S is low e.g. its accuracy on CIFAR-100 is only 14.46% in our experiment, very few images in D_S provide useful knowledge to transfer to the student via the teacher's predictions. Second, since the number of images in D'_S is less than that in D_S, D'_S cannot be used in Equation (4) to train the student network, which requires $|D'_S| = |D_S|$.

4 Experiments and Discussions

We conduct extensive experiments on three benchmark image datasets to evaluate the classification performance (accuracy and F1-score) of our method **KDDM**, compared with strong baselines.

4.1 Datasets

We run experiments on MNIST, CIFAR-10, and CIFAR-100. These datasets are commonly used to evaluate the performance of a KD method [9,5,2,22].

4.2 Baselines

Since there are no existing KD methods dealing with distribution mismatch, we compare **KDDM** with two baselines.

- *Student-Alone*: the student network is trained on the student-data D_S from scratch.
- *Standard-KD*: the student network is trained with the standard KD loss in Equation (1) and with the trade-off factor $\alpha = 0.5$. We also use larger values $\alpha = 0.7$ and $\alpha = 0.9$ to improve the performance of student, as discussed in Section 3.2.

As a reference, we also report the accuracy of current state-of-the-art KD methods, namely KD [9], VID [2], and CRD [22]. Note that these three methods are not comparable with ours since they train both teacher and student networks on the same dataset whereas our method is dealing with the teacher and student networks trained on teacher- and student-datasets with different distributions.

To have a fair comparison, we use the same teacher-student network architecture and hyper-parameters (e.g. batch size and the number of epochs) across all the methods i.e. Student-Alone, Standard-KD, and our **KDDM**. For other baselines KD, VID, and CRD, we obtain their accuracy from related papers[1].

4.3 Results on MNIST

The first experiment shows the KD results on MNIST dataset, where the distribution shift happens between teacher-data D_T and student-data D_S.

Experiment settings. Following [5], we use the HintonNet-1200 network for the teacher, which has two hidden layers of 1,200 units and the HintonNet-800 network for the student, which has two hidden layers of 800 units. These two network architectures are feed-forward neural networks (FNNs), where the student has much fewer parameters than the teacher. We train the teacher network with a batch size of 256 and 20 epochs. We train the student network with a batch size of 64 and 20 epochs. We train the CVAE with FNNs for both encoder and decoder, using a latent dimension of 2, a batch size of 256, and 100 epochs. To find the optimal values (μ^*, σ^*), we use Thomson sampling implemented in the Turbo package [6], with 2,000 iterations, 50 suggestions per batch, and 20 trusted regions. The searching ranges for $\mu = [\mu_1, \mu_2], \sigma = \begin{bmatrix} \sigma_{11} & 0 \\ 0 & \sigma_{22} \end{bmatrix}$ are $\mu_1, \mu_2 \in [-6.0, 6.0]$ and $\sigma_{11}, \sigma_{22} \in [0, 6.0]$. The trade-off β in our new loss function (see Equation (4)) is set to 0.5, and $\beta = 0.5$ is used across all experiments.

The MNIST dataset has 28×28 images from 10 classes ($[0, 1, ..., 9]$), containing 60K training images (called D_{train}) and 10K testing images (called D_{test}). We use D_{train} as the student-data D_S while we create the teacher-data D_T by randomly removing 98% of images in even classes $0, 2, 4, 6, 8$ in D_{train}, leading to two different distributions for D_T and D_S. We train the teacher network on D_T and the student network on D_S, then evaluate both teacher and student networks on *hold-out* D_{test}. The distributions of three datasets D_T, D_S, and D_{test} are illustrated in Figure 3.

Quantitative results. From Table 1(a), we can see our **KDDM** outperforms both Student-Alone and Standard-KD ($\alpha = 0.5$). **KDDM** achieves 98.30%, which is much better than Standard-KD achieving only 96.61%. The teacher's accuracy achieves only 91.52% due to the distribution mismatch between D_T versus D_S and D_{test}, leading to the poor performance of Standard-KD, as discussed

[1] This is possible because we use benchmark datasets, and the training and test splits are fixed.

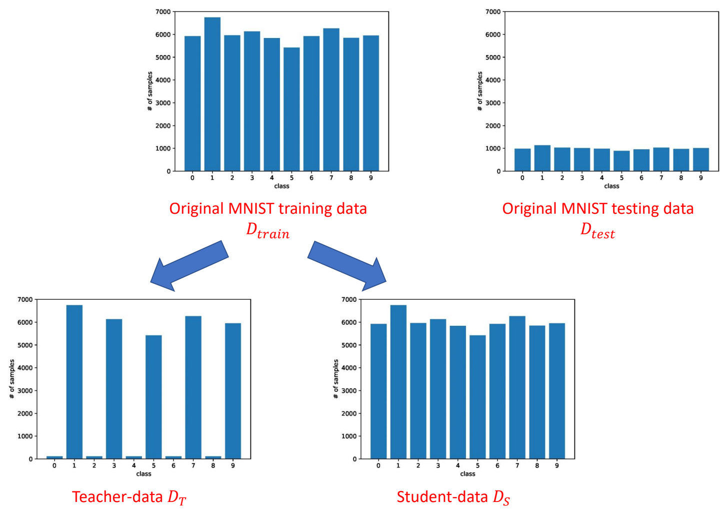

Fig. 3: Distributions of teacher-data D_T, student-data D_S, and test data D_{test}. The distribution of D_T is much different from that of D_S while the distributions of D_S and D_{test} are quite similar (i.e. the number of samples among classes is balanced). *We expect that the teacher trained on D_T will achieve low accuracy on D_S and D_{test} due to the distribution mismatch.*

Table 1: Classification results on MNIST: **(a)** teacher-data D_T and student-data D_S come from two different distributions and **(b)** D_T and D_S are identical. The results of **(b)** are obtained from [5]. "-" means the result is not available in the original paper.

(a) $P(D_T) \neq P(D_S)$

Model	**Accuracy**	**F1-score**
Teacher	91.52%	91.40%
Student-Alone	98.21%	98.19%
Standard-KD ($\alpha = 0.5$)	96.61%	96.60%
Standard-KD ($\alpha = 0.7$)	97.93%	97.92%
Standard-KD ($\alpha = 0.9$)	98.17%	98.15%
KDDM (Ours)	**98.30%**	**98.29%**

(b) $D_T \equiv D_S \equiv D_{train}$

Model	**Accuracy**	**F1-score**
Teacher	98.39%	-
Student-Alone	98.11%	-
KD [9]	98.39%	-
VID [2]	-	-
CRD [22]	-	-

in Section 1. When we increase the value of α to 0.7 and 0.9 (i.e. the student relies more on the true labels of D_S), the accuracy of Standard-KD is improved as expected (see Section 3.2), and approaches the accuracy of Student-Alone.

We also report the accuracy of KD method [9] in Table 1(b) for a reference. KD uses the teacher trained on the original MNIST training data D_{train}, which has more advantages than our **KDDM**. KD's teacher achieves 98.39%, which is much higher than that of our teacher (91.52%). However, our KD framework – **KDDM** still achieves a comparable accuracy with KD (98.30% vs. 98.39%).

4.4 Results on CIFAR-10

The second experiment shows the KD results on CIFAR-10 dataset, where the distribution shift happens between teacher-data D_T and student-data D_S.

Experiment settings. We use ResNet-50 and ResNet-20 models for teacher and student networks respectively. These two network architectures are convolutional neural networks (CNNs), where the teacher has 763K parameters and the student has 274K parameters. We train the teacher and student networks with a batch size of 32 and 200 epochs. We train the CVAE with CNNs for both encoder and decoder, using a latent dimension of 2, a batch size of 64, and 600 epochs. We find the optimal values (μ^*, σ^*) in the same way as in our experiments on MNIST.

The CIFAR-10 dataset is set of 32×32 pixel RGB images with 10 classes, and contains 50K training images (called D_{train}) and 10K testing images (called D_{test}). We use D_{train} as the student-data D_S while we create the teacher-data D_T by randomly removing 98% of images in even classes in D_{train}, leading to $P(D_T) \neq P(D_S)$. We train the teacher network on D_T and the student network on D_S, then evaluate both teacher and student networks on *hold-out* D_{test}.

Quantitative results. From Table 2(a), we can see the accuracy of teacher on D_{test} is only 70.73%, which is much lower than that of Student-Alone trained on D_S. This is because of the distribution mismatch between D_T versus D_S and D_{test}. Consequently, when this poor teacher is used for KD, it makes harmful effects to Standard-KD ($\alpha = 0.5$), where the accuracy significantly drops around 6% from Student-Alone. Even if we use larger values for α to shift the predictions of Standard-KD to the true labels of D_S, its accuracy is only improved to 90.90%, which is still worse than that of Student-Alone. Our **KDDM** achieves 91.54%, which is better than both Student-Alone and Standard-KD baselines, thanks to the new loss function to train the student in Equation (4).

Compared to state-of-the-art KD methods in Table 2(b), **KDDM** is comparable with KD and VID. Note that both KD and VID train teacher and student networks on the same D_{train}, which obtains a much stronger teacher than our method. Their teacher achieves 94.26% whereas our teacher achieves only 70.73% on the test data D_{test}. However, the accuracy of our **KDDM** and those of KD and VID are comparable (91.54% vs. 91.27% and 91.85%).

Table 2: Classification results on CIFAR-10: **(a)** teacher-data D_T and student-data D_S come from two different distributions and **(b)** D_T and D_S are identical. The results of **(b)** are obtained from [2]. "-" means the result is not available in the original paper.

(a) $P(D_T) \neq P(D_S)$

Model	**Accuracy**	**F1-score**
Teacher	70.73%	69.36%
Student-Alone	91.22%	91.20%
Standard-KD ($\alpha = 0.5$)	85.49%	85.60%
Standard-KD ($\alpha = 0.7$)	89.34%	89.40%
Standard-KD ($\alpha = 0.9$)	90.90%	90.91%
KDDM (Ours)	**91.54%**	**91.56%**

(b) $D_T \equiv D_S \equiv D_{train}$

Model	**Accuracy**	**F1-score**
Teacher	94.26%	-
Student-Alone	90.72%	-
KD [9]	91.27%	-
VID [2]	91.85%	-
CRD [22]	-	-

Qualitative results. In Figure 4, we show the original images in student-data D_S versus the images generated by our method corresponding to four true labels "ship", "deer", "bird", and "automobile". We can see our generated images have comparable qualities with original images, where the objects are easily recognized and clearly visualized. These generated images are also correctly classified by the teacher. In contrast, the original images are wrongly predicted by the teacher since they may not be similar to any images in D_T – the training data of teacher. Remind that D_T has much fewer samples in classes 2 ("bird"), 4 ("deer"), and 8 ("ship"). Consequently, the teacher does not observe a diversity of samples and often predicts wrong labels for the images belonging to these three classes. To overcome this problem, our method tries to generate only images best fitting to the teacher (i.e. the images well-classified by the teacher), and use them for the KD process.

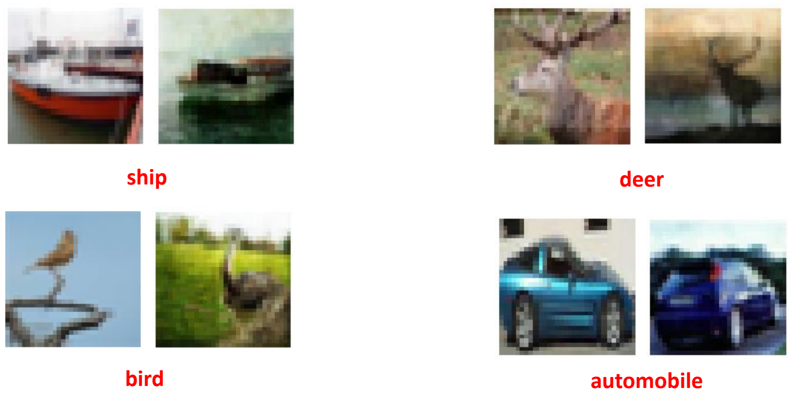

Fig. 4: Original images in original student-data D_S (left) versus generated images in new student-data D'_S (right) in 4 classes "ship", "deer", "bird", "automobile".

4.5 Results on CIFAR-100

The third experiment shows the KD results on CIFAR-100 dataset, where teacher-data D_T and student-data D_S are two different datasets.

Experiment settings. We use a *pre-trained* ResNet-50 model[2] on ImageNet (i.e. teacher-data D_T) for the teacher network and a ResNet-20 model for the student network. We train the student network on the original training set D_{train} of CIFAR-100 (i.e. student-data D_S) with a batch size of 16 and 200 epochs. D_T and D_S differ in their distribution as ImageNet has much higher resolution than CIFAR-100, and two datasets cover different sub-types of a class in general e.g. "green snake" and "grass snake" in ImageNet vs. "snake" in CIFAR-100. We train the CVAE and find the optimal values (μ^*, σ^*) in the same way as in our experiments on CIFAR-10. We evaluate both teacher and student networks on the original testing set D_{test} of CIFAR-100. Note that the classification performance is only measured on 56 overlapped classes between ImageNet and CIFAR-100.

Table 3: Classification results on CIFAR-100: **(a)** teacher-data D_T and student-data D_S are two different datasets and **(b)** D_T and D_S are identical. The results of **(b)** are obtained from [22]. "-" means the result is not available in the original paper.

(a) D_T is ImageNet while D_S is CIFAR-100

Model	**Accuracy**	**F1-score**
Teacher	14.46%	12.70%
Student-Alone	71.09%	71.03%
Standard-KD ($\alpha = 0.5$)	68.21%	68.27%
Standard-KD ($\alpha = 0.7$)	69.57%	69.53%
Standard-KD ($\alpha = 0.9$)	70.75%	70.78%
KDDM (Ours)	**71.14%**	**71.13%**

(b) Both D_T and D_S are CIFAR-100

Model	**Accuracy**	**F1-score**
Teacher	72.34%	-
Student-Alone	69.06%	-
KD [9]	70.66%	-
VID [2]	70.38%	-
CRD [22]	71.16%	-

Quantitative results. From Table 3(a), we can observe similar results as those in the experiments on MNIST and CIFAR-10, where our **KDDM** outperforms both Student-Alone and Standard-KD methods. **KDDM** achieves 71.14% of accuracy, which is better than 71.09% of Student-Alone and 68.21% of Standard-KD ($\alpha = 0.5$). Again, using large values for α helps Standard-KD to improve its performance, but it is still worse than our method. In this experiment, the teacher performs much worse than student models, achieving only 14.46% of accuracy on the testing set of CIFAR-100. This behavior can be explained by the fact that the teacher is pre-trained on ImageNet containing high-resolution images

[2] https://keras.io/api/applications/

(224×224 pixel images) whereas CIFAR-100 only contains low-resolution images (32×32 pixel images). This resolution mismatch makes the teacher difficult to recognize the true objects in CIFAR-100 images [17]. Even using only a very poor performing teacher, **KDDM** still achieves an impressive accuracy.

As a reference, we report the accuracy of three current state-of-the-art KD methods in Table 3(b). Different from our method, these methods train both teacher and student networks on the same training set of CIFAR-100, leading to a good performance for the teacher network (72.34%).

4.6 Distillation when teacher-data and student-data are identical

In this experiment, we show how our method **KDDM** performs when there is no distribution mismatch between teacher-data D_T and student-data D_S, a common setting used in state-of-the-art KD methods. Here, we use the original training set D_{train} of CIFAR-10 for both D_T and D_S.

Table 4 reports the accuracy of our **KDDM** and other baselines on CIFAR-10 when both D_T and D_S are identical. We can see that our teacher improves its accuracy significantly with the full training set of CIFAR-10. It achieves 92.29% of accuracy compared to 70.73% when it was trained on D_T with distribution mismatch (see Table 2(a)). This stronger teacher helps Standard-KD ($\alpha = 0.5$), which boosts its accuracy to 91.50%. **KDDM** can also benefit from this strong teacher, which achieves 91.88% of accuracy slightly better than 91.85% of VID.

Table 4: Classification results on CIFAR-10 when D_T and D_S are identical (i.e. there is no distribution mismatch between D_T and D_S). The accuracy of VID is obtained from [2]. "-" means the result is not available in the original paper.

$D_T \equiv D_S \equiv D_{train}$		
Model	**Accuracy**	**F1-score**
Teacher	92.29%	92.28%
Student-Alone	91.22%	91.20%
Standard-KD ($\alpha = 0.5$)	91.50%	91.48%
VID [2]	91.85%	-
KDDM (Ours)	**91.88%**	**91.86%**

To summarize, the results in Table 4 confirm the real benefit of our method not only for the KD process when the distribution mismatch happens but also for the KD process when both teacher-data and student-data are identical. In both cases, our method is always better than the standard KD method and the student network trained from scratch on the student-data.

5 Conclusion

We have presented **KDDM** – a novel KD method for distilling a large pre-trained *black-box* deep network into a smaller network while maintaining a high accuracy.

Different from existing KD methods, our proposed method is still effective when the teacher-data and the student-data differ in their distribution. **KDDM** uses an efficient framework to generate images well-classified by the teacher network and a new loss function to train the student network. We demonstrate the real benefits of **KDDM** on three standard benchmark image datasets. The empirical results show that **KDDM** outperforms both the standard KD method and the student model trained from scratch. Our accuracy is also comparable with those of state-of-the-art KD methods that assume no distribution mismatch between teacher-data and student-data.

Our future work will study how to extend our method to the setting of *white-box* teacher, which is expected to further improve our student's accuracy since we now can also transfer the representation learned by the teacher to the student.

Acknowledgment

This research was a collaboration between the Commonwealth Australia (represented by Department of Defence) and Deakin University, through a Defence Science Partnerships agreement.

References

1. Adriana, R., Nicolas, B., Ebrahimi, S., Antoine, C., Carlo, G., Yoshua, B.: Fitnets: Hints for thin deep nets. In: ICLR (2015)
2. Ahn, S., Hu, X., Damianou, A., Lawrence, N., Dai, Z.: Variational information distillation for knowledge transfer. In: CVPR. pp. 9163–9171 (2019)
3. Chawla, A., Yin, H., Molchanov, P., Alvarez, J.: Data-Free Knowledge Distillation for Object Detection. In: CVPR. pp. 3289–3298 (2021)
4. Chen, G., Choi, W., Yu, X., Han, T., Chandraker, M.: Learning efficient object detection models with knowledge distillation. In: NIPS. pp. 742–751 (2017)
5. Chen, H., Wang, Y., Xu, C., Yang, Z., Liu, C., Shi, B., Xu, C., Xu, C., Tian, Q.: Data-free learning of student networks. In: ICCV. pp. 3514–3522 (2019)
6. Eriksson, D., Pearce, M., Gardner, J., Turner, R., Poloczek, M.: Scalable Global Optimization via Local Bayesian Optimization. In: NIPS. pp. 5496–5507 (2019)
7. Gou, J., Yu, B., Maybank, S.J., Tao, D.: Knowledge distillation: A survey. arXiv preprint arXiv:2006.05525 (2020)
8. Guo, G., Zhang, N.: A survey on deep learning based face recognition. Computer Vision and Image Understanding **189**, 102805 (2019)
9. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
10. Kim, J., Park, S., Kwak, N.: Paraphrasing complex network: Network compression via factor transfer. In: NIPS. pp. 2760–2769 (2018)
11. Lee, S., Song, B.C.: Graph-based knowledge distillation by multi-head attention network. arXiv preprint arXiv:1907.02226 (2019)
12. Meng, Z., Li, J., Zhao, Y., Gong, Y.: Conditional teacher-student learning. In: ICASSP. pp. 6445–6449. IEEE (2019)
13. Nayak, G.K., Mopuri, K.R., Chakraborty, A.: Effectiveness of Arbitrary Transfer Sets for Data-free Knowledge Distillation. In: CVPR. pp. 1430–1438 (2021)

14. Nguyen, D., Gupta, S., Rana, S., Shilton, A., Venkatesh, S.: Bayesian optimization for categorical and category-specific continuous inputs. In: AAAI. pp. 5256–5263 (2020)
15. Passalis, N., Tzelepi, M., Tefas, A.: Heterogeneous knowledge distillation using information flow modeling. In: CVPR. pp. 2339–2348 (2020)
16. Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, P., Shyu, M.L., Chen, S.C., Iyengar, S.: A survey on deep learning: Algorithms, techniques, and applications. ACM Computing Surveys **51**(5), 1–36 (2018)
17. Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., Madry, A.: Do Adversarially Robust ImageNet Models Transfer Better? In: NIPS. pp. 3533–3545 (2020)
18. Shen, L., Margolies, L., Rothstein, J., Fluder, E., McBride, R., Sieh, W.: Deep learning to improve breast cancer detection on screening mammography. Scientific Reports **9**(1), 1–12 (2019)
19. Snoek, J., Larochelle, H., Adams, R.: Practical bayesian optimization of machine learning algorithms. In: NIPS. pp. 2951–2959 (2012)
20. Sohn, K., Lee, H., Yan, X.: Learning structured output representation using deep conditional generative models. NIPS pp. 3483–3491 (2015)
21. Sreenu, G., Durai, S.: Intelligent video surveillance: a review through deep learning techniques for crowd analysis. Journal of Big Data **6**(1), 1–27 (2019)
22. Tian, Y., Krishnan, D., Isola, P.: Contrastive representation distillation. In: ICLR (2020)
23. Wang, D., Li, Y., Wang, L., Gong, B.: Neural Networks Are More Productive Teachers Than Human Raters: Active Mixup for Data-Efficient Knowledge Distillation from a Blackbox Model. In: CVPR. pp. 1498–1507 (2020)
24. Yim, J., Joo, D., Bae, J., Kim, J.: A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In: CVPR. pp. 4133–4141 (2017)
25. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys **52**(1), 1–38 (2019)