Online Subclass Knowledge Distillation

Maria Tzelepi, Nikolaos Passalis, Anastasios Tefas

Aristotle University of Thessaloniki, Department of Informatics, Greece

Abstract

Knowledge Distillation has been established as a highly promising approach for training compact and faster models by transferring knowledge from more heavyweight and powerful models, so as to satisfy the computation and storage requirements of deploying state-of-the-art deep neural models on embedded systems. However, conventional knowledge distillation requires multiple stages of training rendering it a computationally and memory demanding procedure. In this paper, a novel single-stage self knowledge distillation method is proposed, namely *Online Subclass Knowledge Distillation* (OSKD), that aims at revealing the similarities inside classes, improving the performance of any deep neural model in an online manner. Hence, as opposed to existing online distillation methods, we are able to acquire further knowledge from the model itself, without building multiple identical models or using multiple models to teach each other, rendering the OSKD approach more effective. The experimental evaluation on five datasets indicates that the proposed method enhances the classification performance, while comparison results against existing online distillation methods validate the superiority of the proposed method.

Keywords: Knowledge Distillation, Online Distillation, Subclass Knowledge Distillation, Self Distillation, Deep Neural Networks.

Email addresses: mtzelepi@csd.auth.gr (Maria Tzelepi), passalis@csd.auth.gr (Nikolaos Passalis), tefas@csd.auth.gr (Anastasios Tefas)

1. Introduction

Deep Learning (DL) models (Deng, 2014), have been extensively used during the recent years in order to resolve a wide spectrum of visual analysis tasks, overthrowing previous solutions (Guo et al., 2016; Araque et al., 2017; Redmon & Farhadi, 2017; Graves et al., 2013; Nweke et al., 2018; Do et al., 2019). Generally, DL models owe their outstanding performance to their depth and complexity. This significantly hampers the applicability of state-of-the-art models on devices with limited computational resources, such as embedded systems or mobile phones, reasonably introducing a challenging demand for developing compact yet effective models, diminishing the storage requirements and the computational cost.

Several solutions have been proposed during the recent few years to accomplish this goal (Cheng et al., 2017). For example, considerable research has been performed on developing compact and effective models by design so as to satisfy the memory and computation requirements and at the same time retain accuracy at high levels, (Howard et al., 2017; Zhang et al., 2018a; Sandler et al., 2018; Iandola et al., 2016; Han et al., 2016; Huang et al., 2018). Another line of research includes the parameter pruning, where the redundancy in the parameters of the model is investigated and the complexity of the model is reduced by removing the redundant parameters (Srinivas & Babu, 2015; Molchanov et al., 2017). Similarly, network quantization removes the number of bits for the parameter representation in order to compress the model (Wu et al., 2016; Han et al., 2016). Finally, *Knowledge Distillation* (KD) (also known as *Knowledge Transfer*) (Hinton et al., 2015; Romero et al., 2014; Buciluă et al., 2006; Ba & Caruana, 2014; Chen et al., 2016; Chan et al., 2015; Tang et al., 2016; Passalis & Tefas, 2018; Passalis & Tefas, 2019; Kim et al., 2018) has been emerged as a highly promising approach to settle this issue proposing to transfer the knowledge from one, usually larger, model to a more compact model.

KD methods fall into two broad categories: *online* and *offline* KD. Offline KD refers to the multistage process of training first a heavyweight and complex model, known as teacher, which accomplishes high performance, and then transferring the knowledge to a more compact and faster model, known as student. More specifically, the student model is trained to regress the so-called *soft labels*, produced by softening the output distribution of the teacher model, that is by raising the temperature of the softmax activation function on the output layer of the teacher model. The motivation behind this practice, is that these soft labels, as opposed to the hard labels, can uncover information of the model's generalization mechanism, aiming at implicitly recovering similarities over the data. Amongst KD methods, a distinct subcategory is the so-called *self-distillation*, where the knowledge is transferred from teachers to students of identical capacity (Furlanello et al., 2018; Lan et al., 2018).

Conventional offline KD is a research topic that has been flourishing in the recent years with a broad spectrum of applications ranging from classification (Mirzadeh et al., 2019; Passalis & Tefas, 2018) and semantic segmentation (Liu et al., 2019), to visual question answering (Mun et al., 2018) and top-N recommendation (Pan et al., 2019, 2020a,b; Zhang & He, 2020).

However, offline KD is inherently accompanied by some major limitations. That is, offline distillation requires a two-step sequential training process which cannot be parallelized. As a result, offline distillation often doubles the training time, which could discourage the use of such methods in practice. Thus, another line of research attempts to mitigate these flaws by developing distillation methods that simplify the training pipeline to a single stage. That is, the so-called *online* KD describes the procedure where the teacher and the student networks are trained simultaneously, that is by omitting the stage of pretraining the teacher network. For instance, a recent online KD includes work proposes to train multiple models mutually from each other (Zhang et al., 2018b), while another approach proposes to create ensembles of multiple identical branches of a target network in order to build a strong teacher and distill the knowledge from the teacher to the target network (lan et al., 2018).

It is noteworthy, as explained in lan et al. (2018) online distillation is able to readily scale up and parallelize the training process with virtually no effort and communication overhead, often matching the theoretical speedup (2×). Also, apart from this, online distillation often allows for training more accurate models compared to offline distillation, since at any given time, the *gap* between the student and teacher model will be smaller compared to offline distillation Mirzadeh et al. (2019).

Furthermore, in (Furlanello et al., 2018) it is demonstrated that useful information about the similarities of the samples with the classes can be obtained even by transferring the knowledge through the class probability distribution from a teacher network of identical capacity to student. In addition, in (Ba & Caruana, 2014) it is demonstrated that small networks usually have the same representation capacity as large networks, however they are harder to train, compared to large networks. Therefore, taking these observations into consideration a question that arises is how can someone efficiently train small yet effective networks, deriving additional information beyond the hard labels from the model itself and also in an online manner.

Additionally, motivated by the basic KD intuition that it is useful to maintain the similarities of the data with the other classes instead of simply training with the hard labels, and also by the inherit inefficiency of the conventional KD when the number of classes is limited and hence the information to be transferred is limited, we advocate that inside the classes there are also subclasses that share semantic similarities, and it is also useful to maintain the similarities of the data with the subclasses. That is, the question that arises is how can someone efficiently train small networks with an additional supervision that conveys extra knowledge about the similarities of the data samples with the subclasses from the model itself and also in an online manner.

Thus, in this paper, we propose a novel online self-distillation approach namely *Online Subclass Knowledge Distillation* (OSKD). The intuition of the proposed distillation method considering a probabilistic view of KD is as follows. During the learning process, the probability distribution of the data is transformed layer by layer in a DL model, learning progressively more complex layer representations. Thus, considering a multiclass classification task, the data representations at the output layer of the model are forced by a regular supervised loss to become one-hot representations. However, the process of converting the complex data representations to one-hot representations usually leads to over-fitting and requiring also deeper and more complicated models. Thus, while the conventional KD methodology manifests that it is useful for each sample to maintain the similarities with the other classes, we argue that it is advantageous to maintain the similarities of the subclasses in order to further improve the generalization ability of the model.

More specifically, the proposed online distillation method considers that inside each class there is also a set of subclasses that share semantic similarities (e.g. blue cars, inflatable boats, etc.). Due to the fact that the subclasses inside each class are unknown and as a consequence it is not feasible to pursue a similar approach of softening their distribution as in the traditional KD, our goal is to discover them during the training procedure. To achieve this goal we propose to estimate them using the neighborhood of each sample. That is, we make the assumption that the nearest neighbors, in terms of a similarity metric, of each sample inside a class share the same semantic similarities, and thus they belong to the same subclass.

Therefore, apart from the regular classification objective, an additional distillation objective is introduced, which encourages the data representations to come closer to the nearest representations of the same class. In this way, the subclasses are revealed throughout the training procedure. At the same time, the data representations are forced to move further away from the nearest representations of the other classes, in order to ensure in this way that the distillation objective will prevent the representation entanglement. As is also validated through the conducted experiments, the proposed method is able to derive useful information and progressively uncover more meaningful subclasses throughout the training procedure, since they are driven by the supervised loss. It is, finally, noteworthy that subclass information has been successfully used to improve the accuracy of various learning problems (Nikitidis et al., 2012, 2014; Maronidis et al., 2015), underlining the importance of harnessing subclass information during the training process of powerful, yet prone to over-fitting, DL models.

The main contributions and advantages of the proposed online distillation method can be summarized as follows:

- The proposed OSKD method is the first KD method which aims at deriving additional knowledge by discovering the subclass structure of the data in an online manner and also from the model itself.

- As opposed to the conventional KD methodology which comes with increased training cost both in terms of time and pipeline complexity, the OSKD method is faster and simpler, since it is single-stage online KD method, and thus it is also rendered as more commercially attractive (Anil et al., 2018). The absence of the stage of training first a strong and heavyweight teacher comes also with significant gains in terms of computation and memory cost.

- The proposed method is capable of deriving additional knowledge beyond hard labels from the model itself. Surveying the relevant literature, we can observe that the competitive online distillation methods require multiple copies of the target network to build a strong teacher, or utilize multiple models to train each other in order to derive additional knowledge, leading to multiple times more computationally expensive training procedures. In opposition, the proposed method derive the additional knowledge from the model itself in an online manner, without the need of utilizing multiple models, and hence it has negligible additional computational cost.

- The proposed method, as it is validated through the conducted experiments, is able to derive useful information about the similarities of the data, progressively more reliable throughout the training procedure, since it is driven by the supervised loss. On the contrary, competitive approaches, which for example include the mutual training of multiple students from a different initial condition may only provide restricted additional information.

- The proposed online method does not require fine-tuning any other hyper-parameter, such as the temperature of the softmax activation function, which is in general crucial for obtaining remarkable improvement when applying the conventional distillation approach.

- OSKD method is model agnostic, that is it can be applied to any DL model to improve its performance. In the performed experiments, several architectures have been utilized, varying from simple and lightweight models to deeper ones (e.g. ResNet (He et al., 2016)), considerably improving the classification performance in any considered case.

- Another critical issue for the effectiveness of the conventional KD is the compatibility between the student and the teacher models. That is, the distillation process is not always effective, for example, it has been demonstrated that when the gap between the teacher and the student is large the latter's performance degrades (Mirzadeh et al., 2019). Therefore, the self-nature of the proposed method inherently guarantees the extraction of useful knowledge compatible to the fast student model.

- The OSKD method is capable of removing the dependency on using separate teacher models, which also reduces the required hardware resources by half, going beyond the state-of-the-art. As a result, the proposed method can provide the benefits of distillation without requiring a separate teacher model and increasing the resources/time needed during the training.

- The proposed distillation method can be combined with any other method for developing effective and faster models, e.g. (Zhang et al., 2018a; Sandler et al., 2018).

The rest of the manuscript is structured as follows. Section 2 discusses previous distillation works. The proposed method is presented in Section 3. Subsequently, the experiments performed to evaluate the proposed method are presented in Section 4, and finally, the conclusions are drawn in Section 5.

2. Previous Work

In this Section, recent works in the general area of KT, as well as on online KD, which is more relevant to our work, are presented.

Knowledge Transfer has been extensively studied during the recent years with a wide range of applications (Pan et al., 2018; Liu et al., 2019; Mun et al., 2018; Wang et al., 2018). Firstly in (Buciluă et al., 2006) and then in (Hinton et al., 2015) the idea of distilling the knowledge from a powerful teacher to a weaker student by encouraging the latter to regress the soft labels produced by the teacher by appropriately raising the temperature of the softmax activation function on the output layer of the network, is proposed. The knowledge transfer procedure is also employed for domain adaptation in combination with limited labeled data, in (Tzeng et al., 2015), while, similarly, knowledge is transferred from a Recurrent Neural Network (RNN) model to a small Convolutional Neural Network (CNN) model, in (Chan et al., 2015). Subsequently, differently from the vast majority of relevant approaches where the teacher model is assumed to be weaker than the student model, knowledge from conventional deep neural networks is used to train a RNN model in (Tang et al., 2016).

Also, the idea of KD (Hinton et al., 2015) is expanded to allow for thinner and deeper students, by using not only soft labels but also hints from the teacher's intermediate layers in order to guide the training of the student model, in (Romero et al., 2014). Then, a method where the student model is trained to maintain the same amount of mutual information between the learned representation and a set of labels as the teacher model is proposed in (Passalis & Tefas, 2019), while a method that uses similarity-induced embedding to transfer the knowledge between two layers of neural networks, is proposed in (Passalis & Tefas, 2018). A KD method where the student model is encouraged to mimic the attention map of the teacher model is proposed in (Zagoruyko & Komodakis, 2017), whilst an approach where the parameters of the student model are initialized according to the parameters of the teacher model is proposed in (Chen et al., 2016). Additionally, under the information-theoretic perspective, knowledge transfer is formulated as maximizing the mutual information between the student and the teacher networks in (Ahn et al., 2019). A multi-step KD approach where an intermediate-sized network is utilized to bridge the gap between the student model and the teacher model is proposed in (Mirzadeh et al., 2019), since, as it is stated, the performance degrades when the gap between the teacher model and student model is large.

Furthermore, in a concurrent work (Müller et al., 2020), a subclass knowledge distillation is proposed, where the teacher model is forced to divide each class into subclasses that discovers during the training, and then the student model is trained to match the subclass probabilities. It should be noted, that this work, as opposed to the proposed subclass distillation method, is an offline method, that is, it requires the pretraining step of the teacher model in order to discover the subclasses and after the network's convergence, the student model is trained. Thus, this work is accompanied by the discussed shortcomings of offline distillation methodologies. Furthermore, this work refers to problems with few classes, and focuses attention on binary classification problems. On the contrary, the proposed method is an online distillation method, that is, it discovers the subclasses in a single training step, and also, as it experimentally validated, improves the classification performance for various multiclass problems, ranging from few classes (i.e. 10 classes) to much more classes (i.e. 200 classes).

Surveying the recent literature, several self-distillation approaches have been proposed. Self-distillation as we have previously mentioned describes the kind of distillation where distillation is applied from one model to another of identical architecture. For example, KD is applied from a teacher to a student of identical architecture where the student accomplishes better performance while it is also optimized faster, in (Yim et al., 2017). The flow of solution procedure matrix is utilized in this approach instead of the previously mentioned hints for transferring the knowledge between the intermediate layers. A self-distillation approach where a teacher model is initially trained, and then after its convergence an identical student model is trained with both the goals of the hard labels and matching the output of the teacher model is proposed in (Furlanello et al., 2018), however without softening the logits (i.e. the inputs to the final softmax activation function) by raising the temperature. Similarly, a target model is trained with a conventional supervised loss, the self-discovered knowledge is extracted, and in the second training stage, the model is trained both with the supervised and distillation losses, in (Lan et al., 2018).

In the recent literature, several online distillation works have also been proposed. A method namely co-distillation, improves the performance by proposing to train k copies of a target model in parallel, by adding a distillation term to the loss function of the i-th model to match the average prediction of the other models, in (Anil et al., 2018). A similar approach where multiple students teach each other throughout the training process is proposed in (Zhang et al., 2018b). That is, each student is trained with a conventional supervised learning loss and a distillation loss that matches each student's class posterior probabilities with the class probabilities of other students. In this way, each model acts as a teacher of the other models. In this approach, as opposed to the aforementioned co-distillation method (Anil et al., 2018), different model architectures can be utilized for the mutual training.

Subsequently, an online distillation approach proposes to build a multi-branch version of the network by adding identical branches, each of which constitutes an independent classification model with shared low level layers, and to create a strong teacher model utilizing a gated logit ensemble of the multiple branches in (lan et al., 2018). Each branch is trained with the conventional classification loss and the distillation loss which regresses the teacher's output distributions.

Finally, in a recent work (Kim et al., 2019), the previous works (Zhang et al., 2018b) and (lan et al., 2018) are combined, by proposing an online mutual knowledge distillation method for enhancing both the performance of the fusion module and the sub-networks. That is, when different sub-networks are used, the sub-networks are trained similar to (Zhang et al., 2018b), whilst when identical sub-networks are used, the low level layers are shared, and a multi-branch architecture similar to (lan et al., 2018) is used. The architecture consists of an ensemble classifier using the ensemble logit produced from the sub-networks and a fused classifier, using the fused feature map. The model distills knowledge from the ensemble classifier to the fused classifier, and simultaneously from the fused classifier to each sub-network classifier.

Finally, in Table 1, a summary of the most representative previous works providing information on their key attributes, is presented. The summary Table distinguishes online against offline previous KD works. Provides also information on self-distillation works. Finally, it also distinguishes previous KD works based on the carrier of the knowledge (that is, output layer against intermediate layers).

3. Proposed Method

In this paper, we propose a novel online subclass distillation method which allows for developing efficient and fast-to-execute models for various applications with computational and memory restrictions (e.g. generic robotics applications). Consider for example the problem of crowd detection for autonomous unmanned aerial vehicles (Tzelepi & Tefas, 2019). In such a problem, lightweight models, which should be able to operate on-board (that is on low power GPU) at sufficient speed, are required. Additionally, in this problem the accuracy is as important as speed. Thus, the proposed OSKD method would allow for training efficient lightweight models for addressing these problems. It should also be emphasized that in such a problem with two classes (Crowd vs Non-Crowd) the knowledge to be transferred about the similarity with the other class by a regular distillation method would be limited, thus the OSKD could uncover additional useful knowledge about the subclass similarities for improving the classification performance (e.g. crowds of different density and structure).

An overview of the proposed method's pipeline is provided in Fig. 1. Input

Type	**Methods**
Online Distillation	(Zhang et al., 2018b; lan et al., 2018) (Kim et al., 2019; Anil et al., 2018)
Offline Distillation	(Hinton et al., 2015; Chen et al., 2016), (Chen et al., 2017; Zhang et al., 2019), (Meng et al., 2019; Furlanello et al., 2018), (Romero et al., 2014; Zagoruyko & Komodakis, 2017), (Kim et al., 2018; Passalis & Tefas, 2018), (Heo et al., 2019; Passalis & Tefas, 2019), (Jin et al., 2019; Ahn et al., 2019; Müller et al., 2020)
Self Distillation	(Furlanello et al., 2018; Anil et al., 2018), (lan et al., 2018; Ahn et al., 2019), (Lan et al., 2018; Yim et al., 2017)

Knowledge Source	**Methods**
Output Layer	(Furlanello et al., 2018; Hinton et al., 2015), (Chen et al., 2017; Zhang et al., 2019), (Meng et al., 2019; Zhang et al., 2018b), (lan et al., 2018; Kim et al., 2019), (Anil et al., 2018; Müller et al., 2019), (Ding et al., 2019; Kim & Kim, 2017; Müller et al., 2020)
Intermediate Layers	(Romero et al., 2014; Zagoruyko & Komodakis, 2017), (Kim et al., 2018; Passalis & Tefas, 2018), (Heo et al., 2019; Passalis & Tefas, 2019), (Jin et al., 2019; Ahn et al., 2019)

Table 1: Summary of related KD methods

images are propagated to the network. The soft labels that reveal the subclass similarities are computed as described below, and the model is trained both with the conventional supervised loss and the additional distillation loss, so as to maintain these similarities.

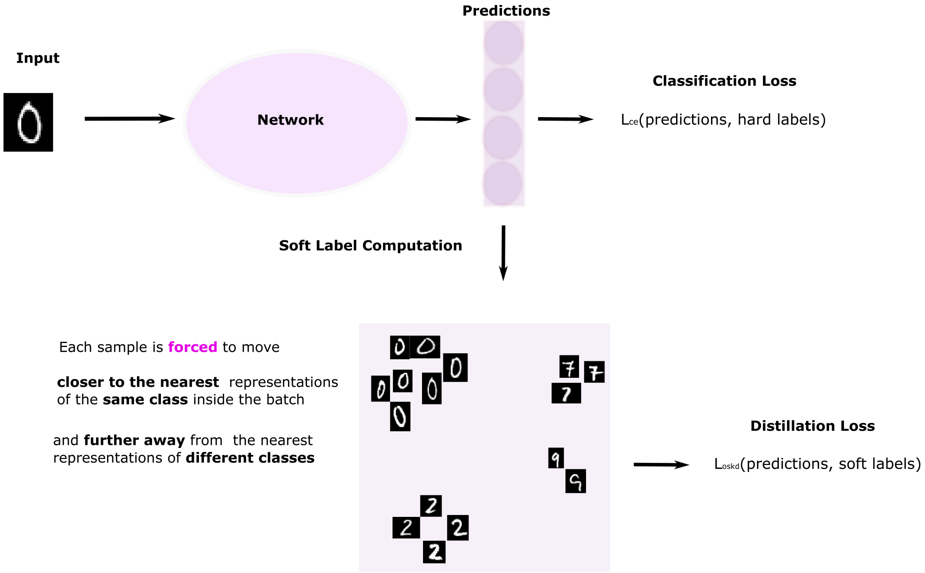

Predictions
Input
Network
Classification Loss
Lce(predictions, hard labels)
Soft Label Computation
Each sample is forced to move
closer to the nearest representations of the same class inside the batch
and further away from the nearest representations of different classes
Distillation Loss
Loskd(predictions, soft labels)

Figure 1: OSKD methodology

An Ω-class classification problem, and the labeled data $\{\boldsymbol{x}_i, \boldsymbol{\omega}_i\}_{i=1}^{N}$, where $\boldsymbol{x}_i \in \Re^D$ an input vector and D its dimensionality, $\boldsymbol{\omega}_i \in \mathcal{Z}^\Omega$ corresponds to its Ω-dimensional one-hot class label vector (hard label) are considered. For an input space $\mathcal{X} \subseteq \Re^D$ and an output space $\mathcal{F} \subseteq \Re^\Omega$, we consider $\psi(\cdot\,; \mathcal{W}) : \mathcal{X} \to \mathcal{F}$ as a deep neural network with $n \in \mathbb{N}$ layers, and set of parameters $\mathcal{W} = \{\boldsymbol{W}_1, \ldots, \boldsymbol{W}_n\}$ where $\boldsymbol{W}_i$ refers to the weights of the i-th layer, which transforms its input vector to a Ω-dimensional vector containing the probabilities for each class. That is, $\psi(\boldsymbol{x}_i\,; \mathcal{W}) \in \mathcal{F}$ corresponds to the output vector of $\boldsymbol{x}_i \in \mathcal{X}$ given by the network ψ with parameters $\mathcal{W}$.

In the typical classification problem, we seek the parameters $\mathcal{W}^*$ that minimize the cross entropy loss, $\mathcal{L}_{ce}$, between the predicted and hard label distributions:

$$\mathcal{W}^* = \arg\min_{\mathcal{W}} \sum_{i=1}^{N} \mathcal{L}_{ce}(\boldsymbol{\omega}_i, \psi(\boldsymbol{x}_i\, ; \mathcal{W})), \tag{1}$$

The cross entropy loss for a set of N samples is formulated as:

$$\mathcal{L}_{ce} = -\sum_{i=1}^{N} \sum_{m=1}^{\Omega} \omega_i^m log(z_i^m), \tag{2}$$

where ω_i^m is the m-th element of $\boldsymbol{\omega}_i$ one-hot label vector, and z_i^m refers to the m-th element of the output of the network:

$$z_i^m = \frac{\exp(\psi(\boldsymbol{x}_i\, ; \mathcal{W})^m)}{\sum_{j=1}^{\Omega} \exp(\psi(\boldsymbol{x}_i\, ; \mathcal{W})^j)}. \tag{3}$$

In this work, we propose to distill additional knowledge online from the model itself throughout the network's training. Towards this end, considering that there are subclasses inside each class that share semantic similarities, we propose to maintain these similarities, which are ignored during the network's training only with the hard labels. Since the subclasses inside each class are unknown, we propose to estimate them using the neighborhood of each sample. That is, we assume that the nearest neighbors, in terms of a similarity metric, of each sample inside a class belong to the same subclass (i.e. share the same semantic similarities).

Thus, for each representation $\psi(\boldsymbol{x}_i\, ; \mathcal{W}) \in \mathcal{F}$ we also define the set $\mathcal{R}^i$ containing the nearest representations, in terms of Euclidean distance, belonging to the same class, $\psi(\boldsymbol{x}_i\, ; \mathcal{W})$, and a set $\mathcal{V}^i$ containing the nearest representations belonging to different classes to the representation. Then, the distillation objective forces the representation to come closer to the nearest neighbors belonging to the same class and moving further away from the nearest representations belonging to different classes. That is, the additional loss for the nearest neighbors of the same class to be minimized is formulated as:

$$\mathcal{L}_1 = \sum_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{R}^i} \|\psi(\boldsymbol{x}_i\, ; \mathcal{W}) - \psi(\boldsymbol{x}_j\, ; \mathcal{W})\|_2^2, \tag{4}$$

and the additional loss for the nearest neighbors of different classes to be maximized is formulated as:

$$\mathcal{L}_2 = \sum_{\boldsymbol{x}_i, \boldsymbol{x}_l \in \mathcal{V}^i} \|\psi(\boldsymbol{x}_i\,; \mathcal{W}) - \psi(\boldsymbol{x}_l\,; \mathcal{W})\|_2^2. \tag{5}$$

It is straightforward to show that equations eq. (4) and (5) can be reformulated as, (Kyperountas et al., 2010):

$$\mathcal{L}_1 = \sum_{\boldsymbol{x}_i \in \mathcal{R}^i} \|\psi(\boldsymbol{x}_i\,; \mathcal{W}) - \boldsymbol{\mu}_r^i)\|_2^2, \tag{6}$$

and

$$\mathcal{L}_2 = \sum_{\boldsymbol{x}_i \in \mathcal{V}^i} \|\psi(\boldsymbol{x}_i\,; \mathcal{W}) - \boldsymbol{\mu}_v^i)\|_2^2 \tag{7}$$

respectively, where $\boldsymbol{\mu}_r^i = \frac{1}{|\mathcal{R}^i|} \sum_{\boldsymbol{x}_j \in \mathcal{R}^i} \psi(\boldsymbol{x}_j\,; \mathcal{W})$, and $\boldsymbol{\mu}_v^i = \frac{1}{|\mathcal{V}^i|} \sum_{\boldsymbol{x}_l \in \mathcal{V}^i} \psi(\boldsymbol{x}_l\,; \mathcal{W})$. Thus, the overall distillation loss is formulating as: $\mathcal{L}_{oskd} = \mathcal{L}_1 + (1 - \mathcal{L}_2)$. Consequently, in the proposed distillation training procedure, we seek for the parameters $\mathcal{W}^*$ that minimize the overall loss of cross entropy, $\mathcal{L}_{ce}$ and distillation, $\mathcal{L}_{oskd}$:

$$\mathcal{W}^* = \arg\min_{\mathcal{W}} \sum_{i=1}^{N} [\mathcal{L}_{ce}(\boldsymbol{\omega}_i, \psi(\boldsymbol{x}_i\,; \mathcal{W})) + \lambda \mathcal{L}_{oskd}(\boldsymbol{\mu}_r^i, \boldsymbol{\mu}_v^i, \psi(\boldsymbol{x}_i\,; \mathcal{W}))], \tag{8}$$

where λ balances the importance between predicting the hard labels and regressing the soft labels.

Simple SGD is utilized to train the model:

$$\Delta \mathcal{W} = -\eta \frac{\vartheta \mathcal{L}}{\vartheta \mathcal{W}}, \tag{9}$$

where $\mathcal{L}$ corresponds to the overall loss. We should note that in our experiments we utilize mini batch gradient descent policy.

In this way, the network concurrently to the cross entropy loss, is trained to match the soft labels enforcing it to regard the similarities inside each class, learning a model which generalizes better.

4. Experiments

First, a toy example, utilizing the MNIST dataset, (LeCun et al., 1998), is constructed so as to illustrate the effect of the proposed distillation method. Subsequently, three datasets were used to evaluate the performance of the proposed method. The descriptions of the datasets as well as the model architecture are presented in the following subsections. We performed four sets of experiments utilizing four different number of nearest neighbors, as well as for two different sizes of mini-batch. An ablation study is also conducted on Cifar-10 dataset in order to validate the effectiveness of subclass knowledge distillation. Test accuracy is used to evaluate the proposed distillation method. Each experiment is executed five times, and the mean value and the standard deviation are reported, considering the maximum value of test accuracy for each experiment. The curves of mean test accuracy are also provided. We also provide the curves of mean test accuracy. Finally, we use the sum of floating point operations (FLOPS) to evaluate the complexity of the proposed OSKD method.

4.1. Datasets

In order to evaluate the performance of the proposed online self-distillation method, we conduct experiments on five datasets.

4.1.1. Cifar-10

The *Cifar-10* dataset, (Krizhevsky & Hinton, 2009), consists of 60,000 images of size 32 × 32 divided into 10 classes with 6,000 images per class. 50,000 images are used as the train set and 10,000 images as the test set. Sample images of the *Cifar-10* dataset are provided in Fig. 2

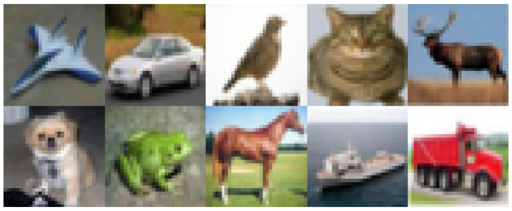


Figure 2: Sample images of the Cifar-10 dataset.

4.1.2. Cifar-100

The *Cifar-100* dataset, (Krizhevsky & Hinton, 2009), consists of 60,000 images of size 32 × 32 divided into 100 classes with 600 images per class. 50,000 images are used as the train set and 10,000 images as the test set.

4.1.3. Street View House Numbers

The Street View House Numbers (SVHN) dataset, (Netzer et al., 2011), is obtained from house numbers in Google Street View images. It contains 73,257 train images and 26,032 test images, divided into 10 classes, 1 for each digit from 0 to 9. Input images are of size 32 × 32 and sample images are provided in Fig. 3

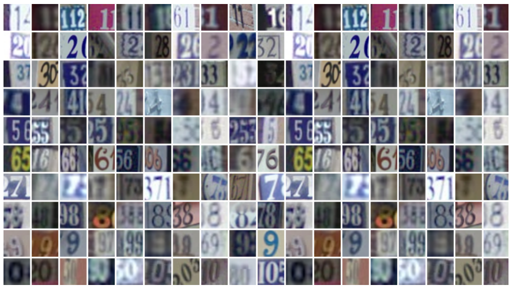


Figure 3: Sample images of the SVHN dataset.

4.1.4. Fashion MNIST

The Fashion MNIST dataset, (Xiao et al., 2017) comprises of 28×28 grayscale images of 70,000 fashion products from 10 categories, with 7,000 images per category. The training set has 60,000 images and the test set has 10,000 images. Sample images are presented in Fig. 4

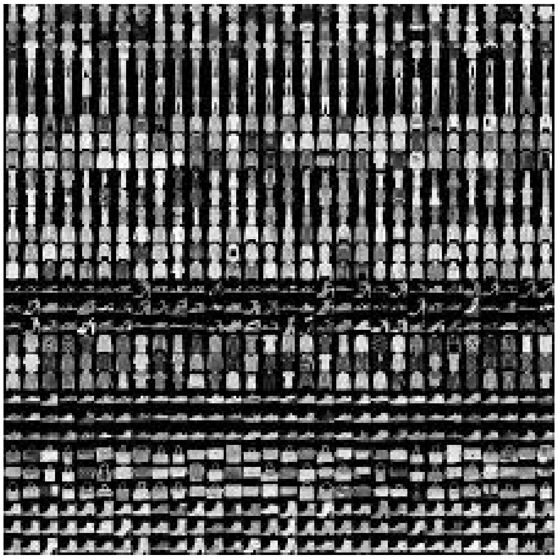


Figure 4: Sample images of the Fashion MNIST dataset.

4.1.5. Tiny ImageNet

The Tiny ImageNet dataset consists of a training set of 200 classes, each containing 500 images, and a validation set consisting of 50 images per class. Input image are of size 64 × 64. Sample images are provided in Fig. 5.

Figure 5: Sample images of the Tiny ImageNet dataset.

4.2. CNN Models

In this work, for the Cifar-10 and SVHN datasets, we utilize a simple CNN model consisting of five layers; two convolutional layers with 6 filters of size

5×5 and 16 filters of size 5×5 respectively, followed by a Rectified Linear Unit (ReLU) (Nair & Hinton, 2010) activation, and three fully connected layers ($128 \times 64 \times 10$). The convolutional layers are followed by a 2×2 max-pooling layer with a stride of 2. In the first two fully connected layers the activation function is the ReLU, while the last output layer is a 10-way softmax layer which produces a distribution over the 10 class labels of the utilized datasets. In the case of the Fashion MNIST dataset, we also utilize a simple architecture consisting of two convolutional with 20 filters of size 5×5 and 50 filters of size 5×5 followed by a ReLU activation, and two fully connected layers (64×10). The convolutional layers are followed by 2×2 max-pooling layer with a stride of 2. In the first fully connected layer a ReLU activation is applied, while the last output layer is a 10-way softmax layer.

In the challenging case of Tiny ImageNet dataset we utilize the common ResNet-50 (He et al., 2016) architecture, without utilizing any pre-trained model in order to avoid image resizing. It is should be emphasized that the target of this work is not to provide state-of-the-art models, but rather to to evaluate the effect of the proposed online subclass distillation method on training lightweight model that can be effectively deployed on embedded and mobile devices. To this aim, we use the aforementioned simple CNN architectures in three out of four cases, while we also use a common powerful network in the case of Tiny ImageNet, validating our claim that the proposed method can be applied to any deep neural model and improve the baseline performance. Finally, for comparison purposes against previous online KD approaches, we also utilize ResNet-32 (He et al., 2016) and Wide ResNet 16-2 (abbreviated as WRN-16-2) (Zagoruyko & Komodakis, 2016) to perform experiments on Cifar-10 and Cifar-100 datasets.

4.3. Implementation Details and Parameter Selection

All the experiments were conducted using the PyTorch framework. The mini-batch gradient descent is used for the networks' training. The learning rate (lr) is set to 10^{-3}, and the momentum is 0.9. All the models are trained on an NVIDIA GeForce GTX 1080 with 8GB of GPU memory for 100 epochs. In order to select the optimal values we performed experiments on the Cifar-10 dataset.

4.3.1. On the lr and the momentum

First, regarding the lr we performed experiments utilizing different values of lr for mini-batch of 64 samples for 100 epochs. The experimental results are illustrated in Fig. 6 and Table 2. As it is shown, the best performance is achieved for $lr = 10^{-3}$. Regarding the momentum, is set to 0.9, since this is the value that is typically used (Sutskever et al., 2013; You et al., 2017).

4.3.2. On the mini batch size and the training epochs

Regarding the mini batch size with regard to the training epochs, we performed experiments using the optimal parameters for mini-batch of size 32, 64, 128, and 256 samples for 200 epochs. Experimental results are presented in Fig. 7 and in Table 3. As we can see better performance is achieved for mini batch of size 32 and 64 samples, while we can see that for these mini batch sizes the models have converged by the first 100 epochs. Thus, for the utilized simple models we use mini batch of 32 and 64 samples for 100 epochs.

4.3.3. On the parameter λ in eq. (8)

In order to select the weight factor λ in eq. (8) for controlling the balance between the contributing losses, we fix the number of nearest neighbors (i.e. we use 4 nearest neighbors) and we conduct experiments for different values of the weight factor λ. The experimental results are presented in Fig. 8. As it can be observed, better results are accomplished for $\lambda = 0.001$, and thus we use this value in the rest experiments. We should finally note that better results could be accomplished through a more extended search for the optimal weight factor.

4.4. Experimental Results

First, a toy example is constructed in order to illustrate the effect of the proposed distillation method. More specifically, we use the MNIST dataset and

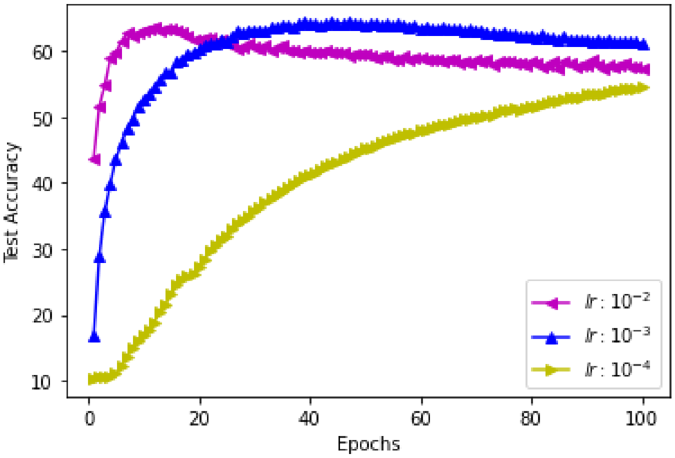


Figure 6: Cifar-10: Mean test accuracy throughout the training epochs for different values of learning rate

LR	**Test Accuracy**
10^{-2}	64.16% ± 0.64%
10^{-3}	**64.73% ± 0.65%**
10^{-4}	54.62% ± 1.05%

Table 2: Cifar-10: Test accuracy for different values of learning rate

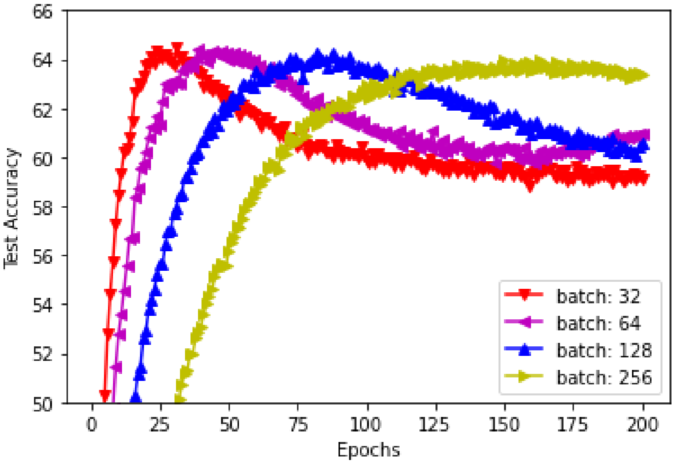


Figure 7: Cifar-10: Mini batch size

Mini batch size	**Test Accuracy**
32	64.85% ± 0.28%
64	64.77% ± 0.44%
128	64.50% ± 0.38%
256	64.30% ± 0.39%

Table 3: Cifar-10: Test accuracy for different mini batch sizes.

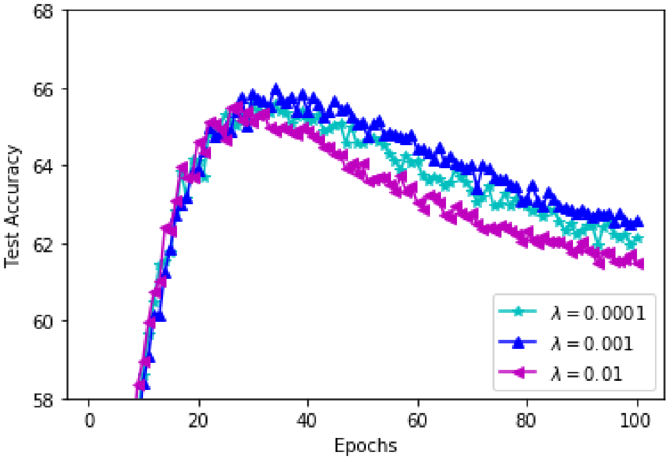


Figure 8: Cifar-10: OSKD weight factor λ in eq. (8)

we build a binary classification problem for discriminating between even and odd digits. For each of the two classes we use three different digits, that is 0, 2, and 4 for the even class, and 1, 3, and 5 for the odd class. The formed train set consists of 36,018 samples, while the formed test set consists of 6,032 samples. In this way, we are enabled to acknowledge in retrospect that there are three distinct subclasses inside each class. Then, we train a simple CNN consisting of two convolutional and two fully connected layers with and without the proposed distillation objective. For the proposed distillation method, we consider 10 nearest neighbors for each sample inside each class, for a mini-batch of 60 samples. Then, we use the t-distributed Stochastic Neighbor Embedding algorithm, (Maaten & Hinton, 2008), to visualize the data representations in the penultimate layer. Experimental results are demonstrated in Figs. 9 and 10 for the train and test sets respectively. As it is shown, the distillation objective achieves to reveal the subclasses inside each class, improving the model's performance both on the train and the test sets.

Subsequently, four sets of experiments performed, for four different numbers of nearest neighbors utilizing also two different mini-batch sizes, in order to validate the proposed online distillation on the Cifar-10, Fashion Mnist, and SVHN datasets. That is, we use 2, 4, 8, and 12 nearest neighbors for each sample (abbreviated as OSKD - 2NN, OSKD - 4NN, OSKD - 8NN, and OSKD - 12NN respectively), and we compare the performance of the proposed method against the baseline performance, that is without distillation (abbreviated as W/o Distillation). The considered mini-batch sizes, are of 32 and 64 samples. The experimental results for mini-batch size of 32 are presented in Table 4, whilst for mini-batch size of 64 are presented in Table 5. Best results are printed in bold. As it can be observed from the reported results, the proposed OSKD method remarkably improves the baseline performance in all the considered cases. We can also observe that better results are reported for 12 nearest neighbors in all the considered cases. Furthermore, we have also conducted individual experiments for larger value of nearest neighbors. That is, we performed experiments for 24 nearest neighbors on the Cifar-10 dataset, achieving accuracy 65.35%±0.58%

for mini batch of 32 samples, which improves the baseline performance (w/o distillation), however is inferior as compared to best performance achieved with 12 nearest neighbors (67.36% ± 0.82%). Correspondingly, for mini batch of 64 samples, accuracy 65.41%±1.09% is accomplished using 24 nearest neighbors, which also improves the baseline performance (w/o distillation) and is inferior as compared to best performance achieved with 12 nearest neighbors (66.26% ± 0.73%).

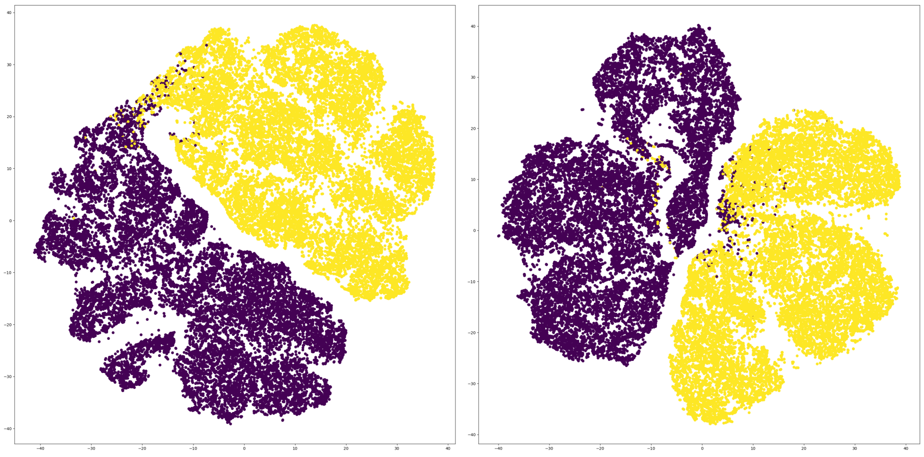
Correspondingly, in Figs. 11-13 the mean test accuracy of the proposed method for the four different number of nearest neighbors against the baseline method is illustrated. The enhanced performance of the proposed method is validated, while the regularization effect of the method is also clearly depicted.

Regarding the Tiny ImageNet dataset, since it differs from the utilized datasets (that is, it consists of 200 classes), we used mini batch 128 samples (and we also consider 24NN for estimating the subclasses). The experimental results are presented in Table 6. As it can be observed, the proposed distillation method achieves to improve the performance on the Tiny ImageNet dataset, too. More meaningful subclasses are discovered using 12NN, leading to the best performance. Consequently, the proposed OSKD method can improve the performance in any considered case, that is, considering 10 class and 200 class problems, and also utilizing simple models, or more powerful ones (ResNet-50).

Method	**Cifar-10**	**SVHN-10**	**Fashion MNIST**
W/o OSKD	64.83% ± 0.57%	88.82% ± 0.22%	91.28% ± 0.14%
OSKD-2NN	66.16% ± 0.76%	89.08% ± 0.26%	91.67% ± 0.13%
OSKD-4NN	66.39% ± 0.77%	89.52% ± 0.23%	91.59% ± 0.07%
OSKD-8NN	66.59% ± 0.78%	89.61% ± 0.29%	91.82% ± 0.08%
OSKD-12NN	**67.36% ± 0.82%**	**89.67% ± 0.28%**	**91.88% ± 0.14%**

Table 4: Test Accuracy - Mini Batch Size: 32

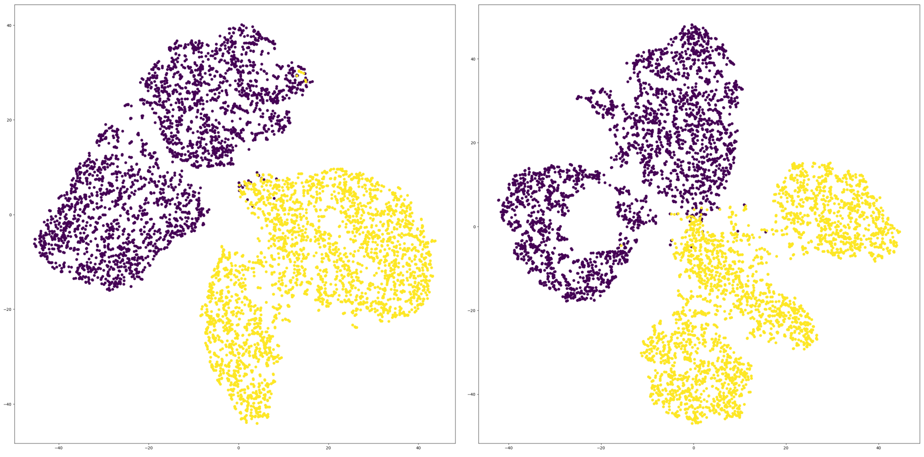
An ablation study is also conducted in order to validate that the effectiveness of the proposed method derives from the subclass knowledge rather than



(a) W/o Distillation

(b) OSKD

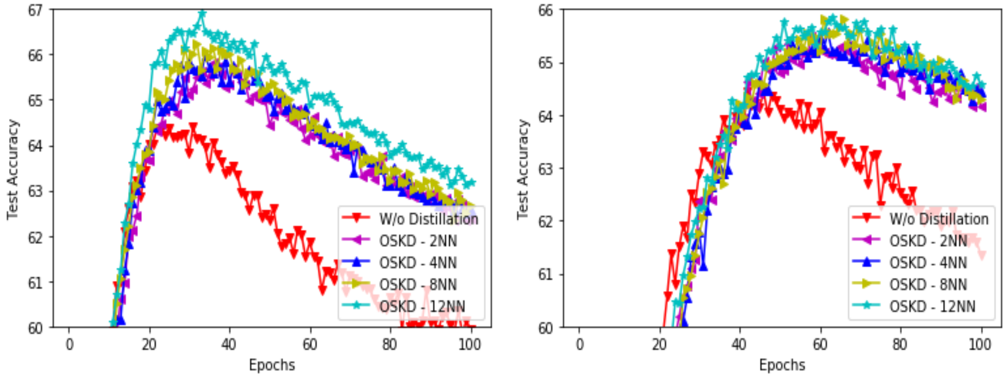
Figure 9: MNIST: Train Set



(a) W/o Distillation

(b) OSKD

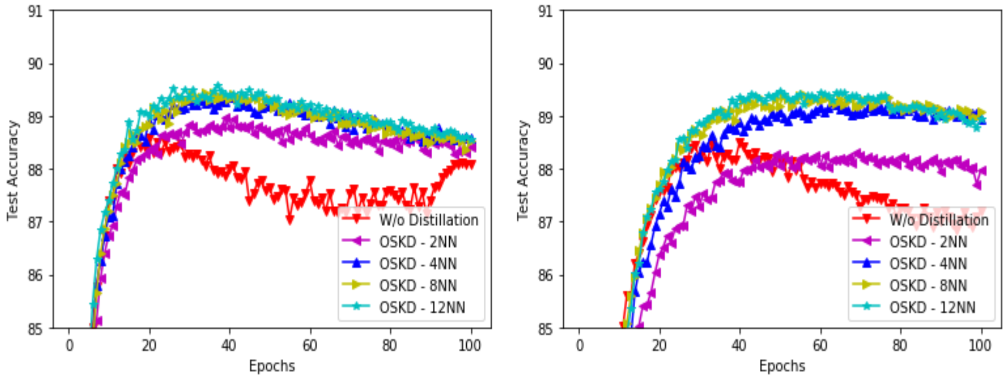
Figure 10: MNIST: Test Set



(a) Batch: 32

(b) Mini-batch: 64

Figure 11: Cifar-10: Test accuracy for different numbers of nearest neighbors inside each class



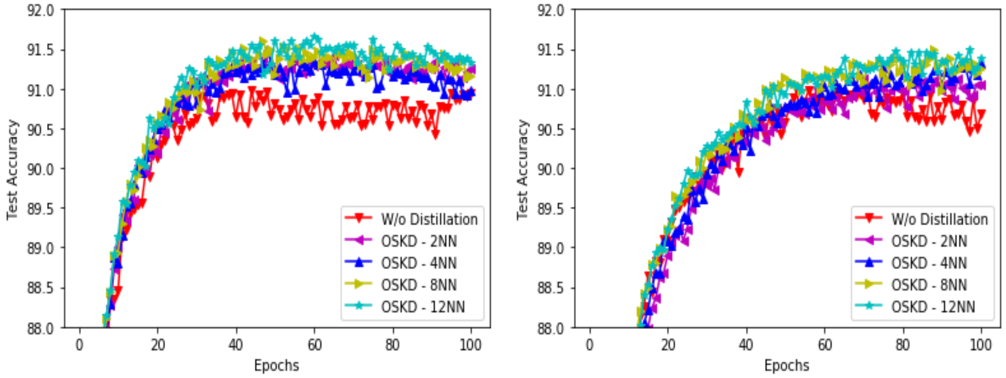
(a) Mini-batch: 32

(b) Mini-batch: 64

Figure 12: SVHN-10: Test accuracy for different numbers of nearest neighbors inside each class

Method	Cifar-10	SVHN-10	Fashion MNIST
W/o OSKD	64.73% ± 0.65%	88.71% ± 0.31%	91.21% ± 0.14%
OSKD-2NN	65.91% ± 0.62%	88.50% ± 0.40%	91.33% ± 0.16%
OSKD-4NN	65.87% ± 0.88%	89.36% ± 0.16%	91.50% ± 0.17%
OSKD-8NN	66.19% ± 0.96%	89.58% ± 0.25%	91.61% ± 0.13%
OSKD-12NN	**66.26% ± 0.73%**	**89.62% ± 0.22%**	**91.68% ± 0.10%**

Table 5: Test Accuracy - Mini Batch Size: 64



(a) Mini-batch: 32

(b) Mini-batch: 64

Figure 13: Fashion MNIST: Test accuracy for different numbers of nearest neighbors inside each class

Method	Tiny ImageNet
W/o OSKD	29.70% ± 0.51%
OSKD-2NN	30.18% ± 0.63%
OSKD-4NN	30.25% ± 0.64%
OSKD-8NN	30.21% ± 0.48%
OSKD-12NN	**30.95% ± 0.43%**
OSKD-24NN	30.02% ± 0.89%

Table 6: Test Accuracy - Mini Batch Size: 128

the additional criterion that forces the data representations of each class to move further away from the nearest representations of the other classes so as to ensure that the distillation objective will not encourage the representation entanglement. To this aim, we conduct experiments utilizing only the subclass knowledge without the aforementioned disentanglement criterion (denoted as Only Class), as well as utilizing only the disentanglement criterion, without the subclass objective (denoted as Only Non-Class). Experimental results on the Cifar-10 dataset are provided in Table 7 considering mini-batch of 32 samples, while experimental results considering mini-batch of 64 samples are provided in Table 8. Regarding the first case of mini-batch of 32 samples, the expected number of samples of the same class is around 4. Indeed, using these numbers of neighbors for estimating the subclasses leads to the best accuracy. On the other hand, the rest of the in-batch samples are expected to belong to a different class, and as a result, using a larger number of neighbors from different classes leads to better accuracy for the disentanglement criterion. On the contrary, considering the second case of mini-batch of 64 samples, where the expected number of samples of the same class is larger, it is shown that the larger numbers of neighbors for estimating the subclasses, that is 8 and marginally 12 nearest neighbors, lead also to the best accuracy. Furthermore, we should highlight that the subclass criterion improves the performance in any case, validating the subclass knowledge hypothesis. Finally, the best performance is accomplished by the combined objective.

NN	**Only Class**	**Only Non-Class**	**Both**
2	65.40% ± 1.18%	65.24% ± 0.84%	**66.16% ± 0.76%**
4	65.95% ± 0.63%	65.72% ± 0.73%	**66.39% ± 0.77%**
8	65.30% ± 0.53%	**66.44% ± 0.45%**	**66.59% ± 0.78%**
12	65.20% ± 0.67%	**66.42% ± 0.67%**	**67.36% ± 0.82%**

Table 7: Cifar-10 - Mini Batch Size: 32 (Baseline: 64.83% ± 0.57%)

Subsequently, since as we have stated the proposed method is model ag-

NN	Only Class	Only Non-Class	Both
2	65.39% ± 1.35%	64.57% ± 0.65%	**65.91% ± 0.62%**
4	65.47% ± 0.30%	65.13% ± 0.50%	**65.87% ± 0.88%**
8	66.12% ± 0.79%	65.90% ± 0.61%	**66.19% ± 0.96%**
12	65.44% ± 0.60%	65.20% ± 0.86%	**66.26% ± 0.73%**

Table 8: Cifar-10 - Mini Batch Size: 64 (Baseline: 64.73% ± 0.65%)

nostic, we have also conducted experiments in order to compare the proposed methods with state-of-the-art distillation methods. More specifically, we utilize two common architectures, that is ResNet-32 (He et al., 2016) and WRN-16-2 (Zagoruyko & Komodakis, 2016), we apply the proposed online distillation method on Cifar-10 and Cifar-100 datasets, and compare the performance with competitive online distillation methods, (lan et al., 2018; Kim et al., 2019; Zhang et al., 2018b), and also with offline distillation methods (KD (Hinton et al., 2015)). We should note some offline distillation methods (e.g. (Romero et al., 2014; Zagoruyko & Komodakis, 2017)) are orthogonal to KD approaches that employ the output of a model as source of knowledge. This category includes most of the online distillation methods proposed in the literature. Therefore, these methods can be combined with any of the proposed online distillation approaches. Therefore, to ensure a fair comparison between the evaluated method we restricted the evaluation to methods that use the output of a model as source of knowledge.

First, for comparing the OSKD method against the online distillation methods (lan et al., 2018; Kim et al., 2019) on Cifar-10 dataset we follow the same training setup as in (lan et al., 2018; Kim et al., 2019) to ensure a fair comparison. That is, for the ResNet-32 case we use the SGD with Nesterov momentum and set the momentum to 0.9. The initial learning rate is set to 0.1 and drops to 0.01 at 50% training and to 0.001 at 75%. The network is trained for 300 epochs. For the WRN-16-2 case, we also use the SGD with Nesterov momentum and set the momentum to 0.9. The initial learning rate is set to 0.1 and drops by 0.2 at

60, 120 and 160 epochs. Models are trained for 200 epochs using mini-batch of 128 samples. We should highlight that for as much as possible fair comparisons, we use only two sub-networks in all the competitive approaches, similar to (Kim et al., 2019), since the proposed method is a single branch method, that is, it does not utilize multiple branches of the network. Thus, we compare the OSKD method with ONE distillation method, considering the average performance of the two branches, and correspondingly with the FFL-S distillation method considering the average performance of the two sub-networks. We should highlight that the number of parameters in both FFL-S and ONE cases in the test phase is identical to the OSKD case, since the additional branches in both cases as well as the fusion module in FFL-S are removed during the test phase.

Furthermore, except for the competitive online distillation methods, we also compare the performance of the proposed method with the ensembling methods, that is ONE-E and FFL, even though we do not follow an ensembling methodology. That is, we also compare the performance of the proposed student model, against the ensemble models which serve as teachers. It should be emphasized that the number of parameters is 0.83M in ONE-E and 0.85M in FFL, while the number of the parameters of OSKD is 0.50M considering ResNet-32, while the number of parameters in ONE-E is 1.24M, and 1.29M in FFL, while the number of parameters of OSKD is 0.70M, considering WRN-16-2.

Evaluation results are presented in Table 9 considering the WRN-16-2, and in Table 10 considering the ResNet-32 model. As we can see from the demonstrated results, the proposed online distillation method achieves superior performance over competitive online distillation methods, as well as over both the ensebmling methods, considering the WRN-16-2 model, and one of them considering the ResNet-32 model.

Subsequently, we performed experiments utilizing the ResNet-32 model for comparing the performance against the DML (Zhang et al., 2018b) method on Cifar-10 and Cifar-100 datasets. For fair comparisons, we use the same experimental setup as in (Zhang et al., 2018b). That is, we use SGD with Nesterov momentum and set the initial learning rate to 0.1, momentum to 0.9

Method	**Test Accuracy**
WRN-16-2	93.55% ± 0.11%
ONE (lan et al., 2018)	93.76%± 0.16%
FFL-S (Kim et al., 2019)	93.79% ± 0.12%
OSKD	**93.96% ± 0.13%**
ONE-E (lan et al., 2018)	93.84%± 0.20%
FFL (Kim et al., 2019)	93.86% ± 0.11%

Table 9: Comparisons against online distillation methods on Cifar-10 utilizing the WRN-16-2 architecture.

Method	**Test Accuracy**
ResNet-32	93.07% ± 0.17%
ONE (lan et al., 2018)	93.76%± 0.12%
FFL-S (Kim et al., 2019)	93.81% ± 0.12%
OSKD	**93.93% ± 0.09%**
ONE-E (lan et al., 2018)	93.93%± 0.17%
FFL (Kim et al., 2019)	**94.02% ± 0.12%**

Table 10: Comparisons against online distillation methods on Cifar-10 utilizing the ResNet-32 architecture.

Method	**Test Accuracy**
ResNet-32	92.47%
KD (Hinton et al., 2015)	92.75%
DML (Zhang et al., 2018b)	92.74% (Net1: 92.68% Net2: 92.80%)
OSKD	**93.30% ± 0.16%**

Table 11: Comparisons against online and offline distillation methods on Cifar-10 utilizing the ResNet-32 architecture.

and mini batch size to 64. The learning rate dropped by 0.1 every 60 epochs and we train for 200 epochs. We also include comparisons against the most common offline KD method (Hinton et al., 2015), using as teacher model the more powerful ResNet-110 model.

It should be highlighted that the experimental setup used in the comparisons against DML and KD, as well as against ONE and FFL, follows the same setup as the one proposed in the literature in order to ensure fair comparisons. As a result, the are slight differences compared to some of the experiments conducted on the same dataset for evaluating the effect of different parameters, for which a more lightweight architecture was employed (all parameters are reported in the corresponding experiments). Experimental results are provided in Table 11 for the Cifar-10 dataset, and in Table 12 for the Cifar-100 dataset. For the DML method we provide the average performance of the two networks and we also provide the performance of the two networks separately. As it is shown the proposed method achieves superior performance over the competitive online method as well as against the offline KD method. We should highlight that DML utilizes a different experimental setup from the previous online distillation methods (Kim et al., 2019; lan et al., 2018). The proposed method achieves accuracy 93.93% on Cifar-10 using their utilized experimental setup, as presented in Table 10.

Finally, we evaluate the complexity of the proposed online distillation method using the sum of floating point operations (FLOPS) in one forward pass on a

Method	**Test Accuracy**
ResNet-32	68.99%
KD (Hinton et al., 2015)	71.17%
DML (Zhang et al., 2018b)	70.97% (Net1: 71.19% Net2: 70.75%)
OSKD	**71.23% ± 0.09%**

Table 12: Comparisons against online and offline distillation methods on Cifar-100 utilizing the ResNet-32 architecture.

fixed input size. Model size, represented by the model's parameters, is also reported for each of the utilized models. We use the ResNet-32 and WRN-16-2 models on the Cifar-10 dataset. In order to highlight the effectiveness of the proposed method we compare the complexity with the most famous offline KD method, (Hinton et al., 2015). In this case, for the ResNet-32 student model, we use as teacher the stronger ResNet-110 model. Correspondingly, for the WRN-16-2 student model, we use as teacher the stronger Wide ResNet 40-2 model (abbreviated as WRN-40-2).

Evaluation results are presented in Table 13. From the demonstrated results, it is validated the proposed distillation method is significantly more efficient as compared to the conventional offline methodology. We should also note that competitive online distillation methods that utilize multiple branches or copies of a given network, require at least two times more FLOPS than the proposed one. That is, the proposed online distillation method is also more efficient as compared to competitive online methods, too. The significant differences that are observed between the proposed method and the KD method is due to the fact that the proposed one does not require keeping and using a separate teacher model for the distillation process. This can be better understood if we use the notation N_T to refer to the FLOPS required for a single feed-forward pass of the teacher and N_S to refer to the the number of FLOPS required for a single feed-forward pass of the student. Then, the complexity of distillation methods that follow the typical separate teacher-student paradigm is $O(k \cdot N_T + N_S)$, where k

Method	**Teacher**	**Student**	**Complexity**
KD (Hinton et al., 2015)	ResNet-110 (1.7M)	ResNet-32 (0.5M)	0.33 GFLOPS
OSKD	-	ResNet-32 (0.5M)	0.07 GFLOPS
KD (Hinton et al., 2015)	WRN-40-2 (2.26M)	WRN-16-2 (0.7M)	0.43 GFLOPS
OSKD	-	WRN-16-2 (0.7M)	0.10 GFLOPS

Table 13: Complexity of the proposed OSKD and KD (Hinton et al., 2015) methods using the sum of floating point operations (FLOPS) in one forward pass on a fixed input size utilizing the Cifar-10 dataset. Model size, represented by the model's parameters, is also reported inside parentheses for each of the utilized models.

is the number of teachers used for the distillation process (typically $k = 1$). The proposed method does not utilize a separate teacher model, since it dynamically mines the knowledge from the same model, performing self-distillation. As a result, this lowers the complexity of distillation to $O(N_S)$, which is consistent with the reported results. This can make the proposed method more appealing for practical applications, since it lowers both the time and cost required for using distillation.

5. Conclusions

In this paper a novel single-stage knowledge distillation method is proposed, namely *Online Subclass Knowledge Distillation*, that aims to reveal the similarities inside classes, improving the performance of any deep neural model in an online manner. As opposed to existing online distillation methods, the proposed method is capable of obtaining further knowledge from the model itself, without building multiple identical models or using multiple models to teach each other, rendering the OSKD method more efficient. The experimental evaluation on five datasets validates efficiency of the proposed method, while comparison results against existing online distillation methods validate the superiority of the proposed method.

5.1. Limitations and future research direction

The proposed method, allows for efficiently lightweight deep learning models, by distilling subclass knowledge from the model itself, without training first a powerful model as in the conventional KD. However, in spite of the discussed advantages over the conventional KD and existing online KD methods, this also may restricts the potentials of the method, since it is based on the features produced by the lightweight model in order to invent the possible subclasses. Thus, despite lightweight models were successfully used in the conducted experiments, more research is needed in order to establish the limits of this process, i.e., how small can a model be in order to produce reliable features that would allow for revealing meaningful subclasses. Furthermore, in this work we utilized the output representations to perform the knowledge distillation, however intermediate representations could also be investigated in future work, since they carry, in general, useful information and have also been utilized in previous works for knowledge transfer, e.g. (Romero et al., 2014).

Acknowledgment

This research is co-financed by Greece and the European Union (European Social Fund - ESF) through the Operational Programme “Human Resources Development, Education and Lifelong Learning 2014-2020” in the context of the project “Lightweight Deep Learning Models for Signal and Information Analysis” (MIS 5047925).

References

Ahn, S., Hu, S. X., Damianou, A., Lawrence, N. D., & Dai, Z. (2019). Variational information distillation for knowledge transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 9163–9171).

Anil, R., Pereyra, G., Passos, A., Ormandi, R., Dahl, G. E., & Hinton, G. E. (2018). Large scale distributed neural network training through online distillation. In *Proceedings of the International Conference on Learning Representations*.

Araque, O., Corcuera-Platas, I., Sánchez-Rada, J. F., & Iglesias, C. A. (2017). Enhancing deep learning sentiment analysis with ensemble techniques in social applications. *Expert Systems with Applications*, *77*, 236–246.

Ba, J., & Caruana, R. (2014). Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems 27* (pp. 2654–2662).

Buciluă, C., Caruana, R., & Niculescu-Mizil, A. (2006). Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* KDD '06.

Chan, W., Ke, N. R., & Lane, I. (2015). Transferring knowledge from a rnn to a dnn. In *Proceedings of the Annual Conference of the International Speech Communication Association* (pp. 3264–3268).

Chen, G., Choi, W., Yu, X., Han, T., & Chandraker, M. (2017). Learning efficient object detection models with knowledge distillation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 742–751).

Chen, T., Goodfellow, I. J., & Shlens, J. (2016). Net2net: Accelerating learning via knowledge transfer. In *Proceedings of the International Conference on Learning Representations*.

Cheng, Y., Wang, D., Zhou, P., & Zhang, T. (2017). A survey of model compression and acceleration for deep neural networks. *CoRR*, *abs/1710.09282*.

Deng, L. (2014). A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing*, *3*, e2.

Ding, Q., Wu, S., Sun, H., Guo, J., & Xia, S.-T. (2019). Adaptive regularization of labels. *arXiv preprint arXiv:1908.05474*, .

Do, H. H., Prasad, P., Maag, A., & Alsadoon, A. (2019). Deep learning for aspect-based sentiment analysis: a comparative review. *Expert Systems with Applications*, *118*, 272–299.

Furlanello, T., Lipton, Z. C., Tschannen, M., Itti, L., & Anandkumar, A. (2018). Born again neural networks. In *Proceedings of the International Conference on Machine Learning* (pp. 1602–1611).

Graves, A., Mohamed, A., & Hinton, G. E. (2013). Speech recognition with deep recurrent neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 6645–6649).

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., & Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, *187*, 27–48.

Han, S., Mao, H., & Dally, W. J. (2016). Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *Proceedings of the International Conference on Learning Representations*.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 770–778).

Heo, B., Lee, M., Yun, S., & Choi, J. Y. (2019). Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence* (pp. 3779–3787). volume 33.

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, .

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, *abs/1704.04861*.

Huang, G., Liu, S., Van der Maaten, L., & Weinberger, K. Q. (2018). Condensenet: An efficient densenet using learned group convolutions. In *Proceed-*

ings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2752–2761).

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡0.5mb model size. *CoRR*, *abs/1602.07360*.

Jin, X., Peng, B., Wu, Y., Liu, Y., Liu, J., Liang, D., Yan, J., & Hu, X. (2019). Knowledge distillation via route constrained optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 1345–1354).

Kim, J., Hyun, M., Chung, I., & Kwak, N. (2019). Feature fusion for online mutual knowledge distillation. *arXiv preprint arXiv:1904.09058*, .

Kim, J., Park, S., & Kwak, N. (2018). Paraphrasing complex network: Network compression via factor transfer. In *Advances in Neural Information Processing Systems* (pp. 2760–2769).

Kim, S., & Kim, H. (2017). Transferring knowledge to smaller network with class-distance loss. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*.

Krizhevsky, A., & Hinton, G. (2009). *Learning multiple layers of features from tiny images*. Technical Report Citeseer.

Kyperountas, M., Tefas, A., & Pitas, I. (2010). Salient feature and reliable classifier selection for facial expression classification. *Pattern Recognition*, *43*, 972–986.

lan, x., Zhu, X., & Gong, S. (2018). Knowledge distillation by on-the-fly native ensemble. In *Advances in Neural Information Processing Systems 31* (pp. 7517–7527).

Lan, X., Zhu, X., & Gong, S. (2018). Self-referenced deep learning. In *Asian Conference on Computer Vision* (pp. 284–300). Springer.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*, 2278–2324.

Liu, Y., Chen, K., Liu, C., Qin, Z., Luo, Z., & Wang, J. (2019). Structured knowledge distillation for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2604–2613).

Maaten, L. v. d., & Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, *9*, 2579–2605.

Maronidis, A., Tefas, A., & Pitas, I. (2015). Subclass graph embedding and a marginal fisher analysis paradigm. *Pattern Recognition*, *48*, 4024–4035.

Meng, Z., Li, J., Zhao, Y., & Gong, Y. (2019). Conditional teacher-student learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 6445–6449). IEEE.

Mirzadeh, S., Farajtabar, M., Li, A., & Ghasemzadeh, H. (2019). Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher. *CoRR*, *abs/1902.03393*.

Molchanov, P., Tyree, S., Karras, T., Aila, T., & Kautz, J. (2017). Pruning convolutional neural networks for resource efficient inference, .

Müller, R., Kornblith, S., & Hinton, G. (2020). Subclass distillation. *arXiv preprint arXiv:2002.03936*, .

Müller, R., Kornblith, S., & Hinton, G. E. (2019). In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada* (pp. 4696–4705).

Mun, J., Lee, K., Shin, J., & Han, B. (2018). Learning to specialize with knowledge distillation for visual question answering. In *Advances in Neural Information Processing Systems* (pp. 8081–8091).

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning* (pp. 807–814).

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning, .

Nikitidis, S., Tefas, A., Nikolaidis, N., & Pitas, I. (2012). Subclass discriminant nonnegative matrix factorization for facial image analysis. *Pattern Recognition*, *45*, 4080–4091.

Nikitidis, S., Tefas, A., & Pitas, I. (2014). Projected gradients for subclass discriminant nonnegative subspace learning. *IEEE transactions on cybernetics*, *44*, 2806–2819.

Nweke, H. F., Teh, Y. W., Al-Garadi, M. A., & Alo, U. R. (2018). Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges. *Expert Systems with Applications*, *105*, 233–261.

Pan, B., Yang, Y., Li, H., Zhao, Z., Zhuang, Y., Cai, D., & He, X. (2018). Macnet: Transferring knowledge from machine comprehension to sequence-to-sequence models. In *Advances in Neural Information Processing Systems* (pp. 6092–6102).

Pan, Y., He, F., & Yu, H. (2019). A novel enhanced collaborative autoencoder with knowledge distillation for top-n recommender systems. *Neurocomputing*, *332*, 137–148.

Pan, Y., He, F., & Yu, H. (2020a). A correlative denoising autoencoder to model social influence for top-n recommender system. *Frontiers of Computer Science*, *14*, 143301.

Pan, Y., He, F., & Yu, H. (2020b). Learning social representations with deep autoencoder for recommender system. *World Wide Web*, (pp. 1–21).

Passalis, N., & Tefas, A. (2018). Learning deep representations with probabilistic knowledge transfer. In *Proceedings of the European Conference on Computer Vision* (pp. 268–284).

Passalis, N., & Tefas, A. (2019). Unsupervised knowledge transfer using similarity embeddings. *IEEE Transactions on Neural Networks and Learning Systems*, *30*, 946–950.

Redmon, J., & Farhadi, A. (2017). Yolo9000: Better, faster, stronger, . (pp. 6517–6525).

Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., & Bengio, Y. (2014). Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, .

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4510–4520).

Srinivas, S., & Babu, R. V. (2015). Data-free parameter pruning for deep neural networks. In *Proceedings of the British Machine Vision Conference* (pp. 1–12).

Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International conference on machine learning* (pp. 1139–1147).

Tang, Z., Wang, D., & Zhang, Z. (2016). Recurrent neural network training with dark knowledge transfer. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 5900–5904).

Tzelepi, M., & Tefas, A. (2019). Graph embedded convolutional neural networks in human crowd detection for drone flight safety. *IEEE Transactions* *on Emerging Topics in Computational Intelligence*, (pp. 1–14). doi:10.1109/TETCI.2019.2897815.

Tzeng, E., Hoffman, J., Darrell, T., & Saenko, K. (2015). Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 4068–4076).

Wang, X., Zhang, R., Sun, Y., & Qi, J. (2018). Kdgan: knowledge distillation with generative adversarial networks. In *Advances in Neural Information Processing Systems* (pp. 775–786).

Wu, J., Leng, C., Wang, Y., Hu, Q., & Cheng, J. (2016). Quantized convolutional neural networks for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4820–4828).

Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, *abs/1708.07747*.

Yim, J., Joo, D., Bae, J., & Kim, J. (2017). A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 7130–7138).

You, Y., Gitman, I., & Ginsburg, B. (2017). Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, .

Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. In *Proceedings of the British Machine Vision Conference* (pp. 87.1–87.12).

Zagoruyko, S., & Komodakis, N. (2017). Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Zhang, F., Zhu, X., & Ye, M. (2019). Fast human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 3517–3526).

Zhang, S., & He, F. (2020). Drcdn: learning deep residual convolutional dehazing networks. *The Visual Computer*, *36*, 1797–1808.

Zhang, X., Zhou, X., Lin, M., & Sun, J. (2018a). Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6848–6856).

Zhang, Y., Xiang, T., Hospedales, T. M., & Lu, H. (2018b). Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4320–4328).