Unsupervised Knowledge Transfer using Similarity Embeddings

Nikolaos Passalis and Anastasios Tefas

***Abstract*—With the advent of deep neural networks there is a growing interest in transferring the knowledge from a large and complex model to a smaller and faster one. In this work, a method for unsupervised knowledge transfer between neural networks is proposed. To the best of our knowledge the proposed method is the first method that utilizes similarity-induced embeddings to transfer the knowledge between any two layers of neural networks regardless of the number of neurons in each of them. This way, the knowledge is transferred without using any lossy dimensionality reduction transformations or requiring any information about the complex model, except for the activations of the layer used for the knowledge transfer. This is in contrast with most existing approaches that only generate soft-targets for training the smaller neural network or directly use the weights of the larger model. The proposed method is evaluated using six image datasets and it is demonstrated, through extensive experiments, that the knowledge of a neural network can be successfully transferred using different kinds of (synthetic or not) data, ranging from cross-domain data to just randomly generated data.**

***Index Terms*—Knowledge Transfer, Unsupervised Learning, Similarity Embeddings, Neural Network Distillation**

I. Introduction

With the advent of deep neural networks [1], [2], there is a growing interest in transferring the *knowledge* from a large and complex model to a smaller and faster one. This process is known in the literature as *model compression* [3], *neural network distillation* [4], or simply *knowledge transfer* (KT) [5]. Usually the knowledge transfer process works by using the large neural network to produce soft-targets that are then used to train the smaller model [4]. These soft-labels implicitly encode the *similarities* between the training samples thus carrying more information than hard binary labels. Matching the soft-labels also acts as a regularizer for the training process allowing a model to achieve higher accuracy than directly training it with the original labels. In this paper the large model, from which the knowledge is transferred, is called *donor* model, while the smaller model, which is trained using the knowledge from the donor model, is called *receiver* model.

The vast majority of neural network knowledge transfer approaches follows, to a greater or lesser degree, the basic distillation idea: a *transfer* set of data is labeled using the donor model and these annotations are used to train the receiver model. However this approach suffers from a significant drawback: it cannot be used to transfer the knowledge between layers of networks with different number of hidden units since there is no way to measure the distance/similarity between feature vectors of different dimensionality. This also implies that the distillation approach cannot be used when the number of targets does not match, e.g., the large network predicts 20 classes, while the smaller one predicts only 10 classes.

These observations lead us to the following questions. Is it possible to extract the information contained in a neural layer without simply performing regression on its output? Can we directly transfer the knowledge between any two layers of two neural networks, regardless their architecture (e.g., number of units, activation functions, etc.)? Furthermore, even though existing knowledge transfer approaches work in an unsupervised fashion, many of them, e.g,. [4], are combined with a supervised loss function in order to train useful networks. Is is possible to perform purely unsupervised knowledge transfer instead of merely using the knowledge transfer as a regularizer?

In this paper the aforementioned problems are addressed by sampling the geometry of the feature space, as induced by the donor model, and then training the receiver model to mimic this geometry using similarity-induced embeddings [6]. To this end, the similarity matrix of the *transfer set* is calculated using the donor model. Then, the receiver model is trained to "follow" the similarities given by the donor model. That way, the geometry of the feature space, as induced by the donor model, is recreated in a space of arbitrary dimensionality using the receiver model. In contrast to other approaches, such as [5], and [7], the proposed method does not require any knowledge of the donor network's architecture (except from its output) and it does not use dimensionality reduction to match the number of units in the receiver model. Instead, it is the first method that supports *directly* transferring the knowledge between layers of different dimensionality.

The proposed method is capable of performing *unsupervised* knowledge transfer using different types of transfer sets, ranging from pure noise to data from other domains. This can be especially useful in cases where the original training data may not be publicly available due to privacy concerns, license restrictions or confidentiality agreements, even though the trained neural network might be freely distributed. After the knowledge transfer, the network can be finetuned using the regular distillation process or a light-weight classifier, e.g., a nearest centroid classifier, can be learned using a very small set of labeled data (3-4 examples per class). Even though the focus of this paper is unsupervised knowledge transfer, the proposed method can be used to regularize the training procedure when labeled data are available. Nonetheless, we demonstrate that even when used for purely unsupervised knowledge transfer it can achieve remarkable results.

Nikolaos Passalis and Anastasios Tefas are with the Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki 54124, Greece. email: passalis@csd.auth.gr, tefas@aiia.csd.auth.gr

The main contribution of this paper is the proposal of a method for knowledge transfer using similarity embeddings. In contrast to other knowledge transfer methods, the proposed method is *model-agnostic*, i.e., it can be used with any neural network architecture. As experimentally demonstrated in Section IV, the proposed method allows for efficiently transferring the knowledge between models, since it fully exploits the information encoded in the similarity between the transfer samples. Note that other methods, such as the neural network distillation [4], also try to implicitly recover this similarity information using heuristics, such as raising the softmax temperature. Six different image datasets are used to evaluate the proposed method, including a large-scale dataset for learning a light-weight model for facial pose estimation that can be deployed on embedded systems with limited computational resources, such as drones.

The rest of the paper is structured as follows. In Section II the related work is presented and compared to the proposed approach. Next, the proposed method is presented in detail in Section III and evaluated in Section IV. Finally, conclusions are drawn and future work is discussed in Section V.

II. Related Work

Transferring the knowledge of a trained neural network into another one is a quite recent research topic, mainly fueled by the growing complexity of deep neural networks and the need to deploy them into mobile and embedded devices with limited computational resources. Most of the proposed methods for transferring the knowledge use soft-labels, i.e., targets generated by the donor model, and then train the receiver model using these pseudo-labels [3], [4], [8], [9], [10]. Among the first attempts for knowledge transfer using soft-labels is the model compression method proposed in [3]. In [4], the previous approach is extended by tuning the temperature of the softmax activation function before producing the soft-labels. It has been shown that this approach can be used to efficiently regularize the smaller network and achieve better generalization than directly training the network using the labels of the training set. The soft-targets can be also used for pre-training a larger network, as in [11]. In [10], the distillation process is used for domain adaptation using sparsely labeled data. A similar approach is also used in [8], to transfer the knowledge from a recurrent neural network (RNN) to a deep neural network. A quite interesting method is also proposed in [9], where the knowledge is transferred from a weaker donor model to a more powerful receiver network. This allows for training the more complex model using fewer labeled data and it also highlights the regularization nature of the distillation process.

All the previous methods use soft-labels generated by the donor model to transfer the knowledge. In [5], a more direct approach is used, where the weights of the donor model are used to initialize the receiver model allowing for faster convergence. Then the receiver model is trained using a regular training dataset. In [7], the receiver network is trained not only using the soft-targets, but also using *hints* from the intermediate layers. Since the size of the receiver model is usually smaller, this is achieved by using a random projection to reduce the dimensionality of the output of the donor model and match the dimensionality of the smaller receiver network.

To the best of our knowledge we propose the first method for unsupervised knowledge transfer using similarity embeddings that is able to natively handle knowledge transfer between layers of different dimensionality. Note that in contrast to the previously used approaches the method proposed in this paper does not require having access to the actual weights of the network (and/or using specific activation functions), as in [5], or learning low dimension projections, as in [7]. Instead, a similarity embedding is used to transfer the knowledge from the donor model to the receiver model, regardless the actual dimensionality of the layers involved. Note, that similarity embeddings have been used with great success for developing a wide range of dimensionality reduction techniques [6]. However, this is the first time that they are used to transfer knowledge between different neural networks and not merely perform dimensionality reduction.

Transfer learning and domain adaptation techniques, e.g., [10], [12], [13], [14], [15], are also related to KT methods. However, in these techniques the aim is to provide a model that can withstand input distribution changes and/or transfer the knowledge across different domains and tasks. On the other hand, KT techniques mainly aim to efficiently *distill* the knowledge contained in a large and complex network into a smaller one.

III. Proposed Method

Let D be the *donor* network from which the knowledge will be transferred to the *receiver* network R. The output of the i-th layer of each network is denoted by $D(\mathbf{x}, i)$ and $R(\mathbf{x}, i)$ respectively, where $\mathbf{x} \in \mathbb{R}^L$ is an input vector (or tensor) and L is the input dimensionality. Even though both networks must receive input vectors of the same dimensionality there is no constraint on the dimensionality of the next layers. For example, the donor network might be a $10 \times 100 \times 30 \times 15$ MLP, while the receiver network a $10 \times 50 \times 10$ MLP. Also, let $\mathcal{X}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ be the transfer set that is used to transfer the knowledge from the donor network to the receiver network. The transfer set may contain the original training set, i.e., the set used for training the donor network, data from a relevant domain, synthetic data, or just randomly generated data vectors.

The proposed method aims to transfer the knowledge encoded in the m-th layer of the donor network to the l-th layer of the receiver model. To this end, the receiver model is trained to recreate the pairwise similarities between the donor's representations of the points in the transfer set. Let $\mathbf{t}_i = D(\mathbf{x}_i, m) \in \mathbb{R}^{L_m}$ and $\mathbf{y}_i = R(\mathbf{x}_i, l) \in \mathbb{R}^{L'_l}$ be the output of the m-th layer of the donor network and the output of the l-th layer of the receiver model respectively. Note that in general the knowledge might be transferred between different layers ($m \neq l$) of networks with different architecture ($L_m \neq L'_l$).

Let $[\mathbf{T}]_{ij}$ be the similarity between the representations of the i-th and the j-th point of the transfer set, as encoded by a layer of the donor model. This similarity can be estimated using

any similarity metric, such as the cosine similarity $[\mathbf{T}]_{ij} = \frac{\mathbf{t}_i^T \mathbf{t}_j}{||\mathbf{t}_i||_2 ||\mathbf{t}_j||_2}$, or a euclidean distance based metric, e.g., the heat kernel $[\mathbf{T}]_{ij} = exp(-\frac{||\mathbf{t}_i - \mathbf{t}_j||_2}{\sigma})$. In this work, the similarity is calculated using the linear kernel, i.e., the dot-product between the vectors $\mathbf{t}_i$ and $\mathbf{t}_j$:

$$[\mathbf{T}]_{ij} = |\mathbf{t}_i^T \mathbf{t}_j| \tag{1}$$

where the absolute value function is used to ensure that Eq. (1) is a proper similarity metric. Using the linear kernel allows for simpler and faster implementations than other similarity metrics, e.g., the heat kernel involves the computationally intensive calculations of the exponential function, without significantly reducing the quality of the knowledge transfer. Before calculating the similarity matrix $\mathbf{T}$ the values $\mathbf{t}_i$ are normalized using min-max scaling to the range $0 \dots 1$.

The receiver model is then trained to "mimic" the similarity given by the donor model. To this end, the similarity between the representations $\mathbf{y}_i$ and $\mathbf{y}_j$ using the receiver model are similarly defined as:

$$[\mathbf{P}]_{ij} = |\mathbf{y}_i^T \mathbf{y}_j| \tag{2}$$

The receiver similarity matrix, as given in Eq. (2), must closely approximate the donor similarity matrix (given in Eq. (1)). To achieve this goal, the mean squared loss between the target similarity and the actual similarity is used to train the network:

$$J = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} ([\mathbf{T}]_{ij} - [\mathbf{P}]_{ij})^2 \tag{3}$$

where N is the number of samples in the transfer set. Mimicking the pairwise similarities acts as a way to recreate the geometry of the donor model using the receiver model. As it is experimentally demonstrated in Section IV, this allows for efficiently transferring the knowledge between models, since the similarity between different samples encodes more information than a hard binary label. Note that the actual learned feature space might be rotated and distorted during this process. Therefore, if the proposed method is used for knowledge transfer between the final classification layers of two networks, then the original class mapping can be recovered using a lightweight classifier, such as the nearest centroid classifier.

The receiver model can be trained using simple gradient descent:

$$\Delta \mathbf{W} = -\eta \frac{\partial J}{\partial \mathbf{W}} \tag{4}$$

where $\mathbf{W}$ is the matrix of the parameters of the receiver model. The loss derivative is calculated as $\frac{\partial J}{\partial \mathbf{W}} = -\frac{2}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} ([\mathbf{T}]_{ij} - [\mathbf{P}]_{ij}) sign(\mathbf{y}_i^T \mathbf{y}_j)(\frac{\partial \mathbf{y}_i^T}{\partial \mathbf{W}} \mathbf{y}_j + \mathbf{y}_i^T \frac{\partial \mathbf{y}_j}{\partial \mathbf{W}})$, where the derivative $\frac{\partial \mathbf{y}_i}{\partial \mathbf{W}}$ is used to backpropagate the gradients to the receiver model and $sign(\cdot)$ is the sign function (with $sign(0) = 0$). Instead of using the whole transfer set to calculate the loss function, as in Eq. (3), smaller batches are used. The data are shuffled between the training epochs to ensure that different data points are used to calculate the pairwise similarities in each batch. The complete knowledge transfer algorithm is shown in Figure 1.

Input: A transfer set $\mathcal{X}_{train} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ of N training samples and the donor model D
Hyper-parameters: The number of iterations N_{iters}, the learning rate η, the batch size N_{batch} and the layers used for the knowledge transfer (m and l)
Output: The trained receiver model R

procedure KNOWLEDGE TRANSFER
  Initialize the receiver model R
  Initialize the donor scaler (minimum and the maximum values of the m-th layer)
  for $i \leftarrow 1; i \leq N_{iters}; i++$ **do**
    Shuffle the transfer set X_{train}
    for $X_{batch} \in X_{train}$ **do**
      Calculate and scale the activations $\mathbf{t}_i$ of the m-th layer using the donor model
      Update the receiver model R using Eq. (4)
  return the optimized receiver model R

Fig. 1: SKT Algorithm

IV. EXPERIMENTS

Six different image datasets were used to evaluate the proposed method. Four of them, the MNIST dataset [16], the CIFAR10 [17] dataset, the Annotated Facial Landmarks (ALFW) dataset [18], and the Tiny Imagenet dataset[19], were used for both transferring the knowledge and evaluating the quality of the learned models, while the other two, the notMNIST [20], and the CIFAR100 [17], were only used as cross-domain transfer sets.

The ALFW dataset was used for solving a pose estimation task (three class classification: left (yaw less that -10 degrees), center (yaw between -10 and 10 degrees) and right (yaw greater than 10 degrees)). The 75% of the images were used to train the models, while the rest 25% for evaluating the accuracy of the models. The face images were cropped according to the given annotation and then resized to 32×32 pixels. Face images smaller than 16×16 pixels were not used for training or evaluating the model. For the rest of the datasets, the default training/test/validation splits were used. The images of the Tiny Imagenet were resized to 56×56 pixels and data augmentation techniques (random flip with $p = 0.5$, random rotation up to 5 degrees and random crops with padding up to 4 pixels) were also used.

The proposed method is evaluated using the following three experimental setups: a) KT using noise (the transfer set is composed solely of randomly generated data without using any prior information about the actual distribution of the training data), b) KT using domain data (the transfer set is composed of a subset of the training data) and c) KT using cross-domain data (the transfer set is composed of data of a similar, but irrelevant to the classification task, domain.)

After the knowledge transfer, it is impossible to directly evaluate the quality of the model, since the embedded feature space may have been transformed, e.g., rotated. To evaluate the quality of the knowledge transfer we use two different approaches: a) a lightweight nearest centroid classifier (abbreviated as "NCC") is trained on the learned representation using only few training samples (3 samples per class are used in the

Transfer Set	Distill. NCC / NN	SKT NCC	SKT+Distill. NCC / NN
Noise $\mathcal{N}(0.5, 0.25)$	24.78 / 89.96	36.10	**20.44** / 88.77
MNIST(30)	37.87 / 27.10	27.29	27.21 / **26.77**
MNIST(30)+$\mathcal{N}(0, 0.25)$	32.63 / 25.48	28.06	27.57 / **20.23**
MNIST(F)	17.83 / 0.91	12.29	10.82 / **0.80**
notMNIST	19.84 / 4.16	17.90	14.58 / **3.40**
notMNIST+MNIST	16.98 / 0.86	11.98	11.23 / **0.86**

TABLE I: MNIST Evaluation: Knowledge transfer using different transfer sets (classification error rate (%))

conducted experiments) and b) the distillation approach [4] is used to finetune the network towards classification and the output layer is used for the evaluation (abbreviated as "NN"). All the methods were used in a purely unsupervised setting, i.e., no labels were used during the training. The distillation approach was also used in a purely unsupervised setting, i.e., soft-labels were generated in an fully unsupervised fashion by the donor network and then used for training the receiver network without using any hard labels from the dataset (the receiver network is trained to follow the mapping between output neurons and classes, as it is defined by the donor network).

Training Setup: For all the evaluated datasets (except for the Tiny Imagenet) the Adam algorithm [21], was used for the optimization, since it generally provides faster and more stable convergence. The batch size was set to 64, the learning rate to 0.001, and the default parameters of the Adam algorithm were used ($\beta_1 = 0.9, \beta_2 = 0.999$ and $\epsilon = 10^{-8}$) [21]. The knowledge transfer techniques (both the proposed and the distillation) ran for 50 epochs (unless otherwise stated). When noise is used for training, the same amount of synthetic samples as the original training set are generated. For the Tiny Imagenet dataset, the learning rate was set to 0.0001, and the feature vectors used for the SKT were normalized to have unit l^2 norm. For the distillation approach, the softmax temperature was raised to $T = 10$ ($T = 2$ for the Imagenet dataset).

a) MNIST Evaluation: First, the proposed method is evaluated using the well-known MNIST dataset. The donor model is composed of 2 pairs of convolutional layers (64 filters of size 5×5) and 2×2 max pooling layers, followed by two fully connected layers (512×10). The receiver model follows the same architecture, but it has over 30 times less parameters than the donor model (20k vs. 634k), since it only uses 8 filters of size 5×5 for its convolutional layers and a hidden layer with 128 units. The rectifier activation function is used for both networks and dropout is utilized for training the models [22]. When the donor model is trained using the full training split of the MNIST dataset it achieves a test error of 0.64%. On the other hand, the receiver model achieves a test error of 1.17% when trained using the full training dataset, and a test error of 28.33% when trained using 3 randomly sampled training images per class (a total of 30 training images are used, which is the same amount of data used for training the nearest centroid classifier in the subsequent experiments).

The experimental results are shown in Table I. Two classification error rates are reported: the first one is the nearest centroid classifier classification error using the features extracted from the last convolutional layer (abbreviated as NCC), while the second one is the classification error of the output of the network (abbreviated as NN). The proposed approach is abbreviated as SKT (Similarity-based Knowledge Transfer). Since the SKT method is only used to transfer the knowledge of the convolutional layers, only the NCC error is reported. The proposed method is also compared to the distillation approach [4] (left column of Table I). Finally, after performing knowledge transfer using the proposed SKT method, the network can be further finetuned using the distillation approach (right column of Table I, both the NNC and the NN classification error rates are reported).

First, randomly generated data (noise) are used as the transfer set to train the receiver model using the donor model. The notation "Noise $\mathcal{N}(\mu, \sigma^2)$" is used to refer to Gaussian noise with mean μ and standard deviation σ. Even though the receiver network has never seen a real digit during the knowledge transfer (only randomly generated data are used) it achieves a remarkable 20.44% classification error when the NCC classifier is used (over 4% less than using the distillation approach alone). This result is actually better than directly training the network with 30 training samples (28.33% error).

Next, the original training dataset is used for knowledge transfer. Two different setups are used: the first one only uses a limited training set of 30 training samples (30), while the second one uses the full training dataset (F). Using just the 30 training samples leads to a relatively large classification error. However, when the data are augmented by adding Gaussian noise ("MNIST(30)+$\mathcal{N}(0, 25)$"), the classification error drops from 26.77% to 20.23%. In both cases it is evident how the proposed SKT method improves the learned representation over the distillation approach. The NCC classification error decreases by more than 10% when the proposed method is used. When the full training dataset is used (MNIST (F)) the proposed method improves the NCC classification error by 7% over the distillation approach, while reaching a spectacular 0.80% classification error. Therefore, the network trained using the proposed unsupervised SKT method performs better than the network that was directly trained with the labeled training dataset (1.17% classification error).

Finally, cross-domain data from the notMNIST dataset are used for the knowledge transfer. Using the notMNIST dataset allows for achieving a remarkable error of 3.40% (the network has never seen a digit during the training process). Again, the proposed approach leads to better results than the distillation method. Note that only the notMNIST images were used for the SKT and the distillation process. Combining the notMNIST and the MNIST datasets further reduces the classification error to 0.86%.

b) CIFAR10 Evaluation: Next, the CIFAR10 dataset is used to evaluate the proposed method for knowledge transfer. The donor model is composed of six layers, two convolutional layers with 32 filters of size 3×3, one 2×2 max pooling layer, 2 convolutional layers with 64 filters of size 3×3 and a second 2×2 max pooling layer, followed by two fully connected layers (512×10). Local response normalization [23], and dropout [22], were used for the training of the donor and the baseline receiver models. The receiver model

Transfer Set	Distill. NCC/NN	SKT NCC	SKT+Distill. NCC/NN
Noise $\mathcal{N}(0.5, 0.25)$	77.80 / 88.85	78.32	**76.81** / 88.45
CIFAR-10(30)	84.17 / 80.28	78.32	77.56 / **75.57**
CIFAR-10(30)+$\mathcal{N}(0, 0.01)$	83.10 / 79.55	78.48	78.35 / **74.59**
CIFAR-10(F)	74.64 / 31.38	69.59	71.96 / **27.50**
CIFAR-100	73.71 / 35.42	69.83	71.54 / **33.94**
CIFAR-100+CIFAR-10	72.42 / 27.00	67.91	70.36 / **25.69**

TABLE II: CIFAR Evaluation: Knowledge transfer using different transfer sets (classification error rate (%))

uses a simpler architecture: only two convolutional layer are used instead of four (one convolutional layer is removed before each of the max pooling layers) and the final fully connected layer has less hidden units (128×10 instead of 512×10). When the donor model is trained using the full training split of the CIFAR10 dataset it achieves a test error of 20.02%. On the other hand, the receiver model achieves a test error of 28.31% when trained using the full training dataset, and a test error of 82.48% when trained using 3 randomly sampled training images per class. Note the severe degeneration of the receiver model when trained using just 30 training samples. The knowledge (SKT) is transferred between the last convolutional layers of the networks.

As in the MNIST evaluation, the proposed method is first evaluated using randomly generated data as the transfer set. The proposed knowledge transfer method, when combined with finetuning, improves the classification results over the simple distillation process. Surprisingly the receiver model trained using only noise as the transfer set achieves better classification error (76.81%) than the receiver model that was trained using a subsample of the actual training data (82.48%).

When using only 30 training (domain) samples for the knowledge transfer, the model achieves significantly better classification error than the corresponding baseline model (75.57% vs. 82.48%). This is mainly due to the proposed SKT transfer method, since the distillation method achieves 80.28% classification error instead of 75.57%. When noise is added to the data the error of the model is further reduced to 74.59%. When the whole training dataset is used the model achieves a classification error of 27.50% (with finetuning), which is better than the baseline model trained on the same data (28.31%). Again, the proposed method greatly outperforms the distillation process and allows for improving the quality of the knowledge transfer. Finally, the proposed SKT method is evaluated using cross-domain data from the CIFAR100 dataset leading to 33.94% classification error (35.42% when only the distillation process is used). When both the CIFAR100 and the CIFAR10 datasets are used as the transfer set the classification error is further reduced to 25.69%. Again, this is better than directly training the baseline model with the labeled dataset (28.31% classification error).

c) ALFW Evaluation: The proposed SKT method is also evaluated using the AFLW dataset for transferring the knowledge into a lightweight neural network that can be deployed on embedded devices. The donor network is composed of 2 convolutional layers with 16 3×3 filters, followed by a 2×2 max pooling layer, another 2 convolutional layers with

Method	Accuracy (NCC)	Accuracy (NN)
Distillation	78.21%	81.76%
SKT	77.99%	-
SKT + Distill.	**80.81%**	**83.11%**

TABLE III: AFLW Evaluation

Method	top-1 accuracy	top-5 accuracy
Baseline	37.05%	64.00%
Distillation	39.75%	66.73%
SKT + Distill.	**40.42%**	**67.40%**

TABLE IV: Tiny Imagenet Evaluation

32 3×3 filters, a 2×2 max pooling layer and 128×3 fully connected layers. The receiver model is composed of one 3×3 convolutional layer with 8 filters, a 3×3 max pooling layer, another one 3×3 convolutional layer with 16 filters, followed by a 3×3 max pooling layer and 16×3 fully connected layers. Local contrast normalization is used after each pooling layer, rectifier activation functions are used for all the layers (except for the output layer where the softmax activation function is used) and the dropout technique ($p = 0.5$) was used before the fully connected layers. The receiver model is significantly simpler than the donor model reducing the number of used parameters by almost two orders of magnitude (the donor model requires more than 110k parameters, while the receiver model less than 2k parameters). The networks were trained using the cross-entropy loss for 50,000 iterations (batch size of 32 samples) achieving pose estimation accuracy of 87.19% for the donor model and 82.83% for the receiver model.

The experimental results using the proposed method are shown in Table III. For both the distillation and the SKT methods 50,000 training iterations were used (batch size of 32). For the SKT method the knowledge is transferred from the first fully connected layer of the donor to the first fully connected layer of receiver. The proposed SKT + Distillation method increases the pose estimation accuracy over 2.5% for the NCC classifier and over 1.3% (when compared to the plain distillation method). Again, note that this result is better than directly training the receiver network with the available labels highlighting the effectiveness of the proposed approach, even though no labeled samples were used for this process.

d) Tiny ImageNet Evaluation: The proposed method was also evaluated on a more challenging dataset, the Tiny ImageNet dataset [19]. A larger donor network pre-trained on the whole Imagenet dataset, the ResNet-18 [24], was used. On the other hand, a well established lightweight model, the SqueezeNet v.1.1 [25], was used as the receiver network. After finetuning both models on the used dataset (5 epochs for the fully connected layer, followed by 150 full optimization epochs with learning rate decay ($0.001 \rightarrow 0.0001$)), they achieved 61.74% top-1 and 83.97% top-5 accuracy for the ResNet-18, and 37.05% top-1 and 64% top-5 accuracy for the SqueezeNet. The experimental results are shown in Table IV. For the conducted experiments the finetuned SqueezeNet was used to initialize all the models and ensure a fair comparison between the evaluated methods. The knowledge was transferred between the penultimate layers of the networks. Again,

both the distillation and the proposed approach outperform the baseline SqueezeNet demonstrating that even purely unsupervised training can increase the accuracy of the models. Also, the proposed approach (combined with distillation) further increases the accuracy to 40.42% top-1 (67.40% top-5).

V. Conclusions

In this paper, a method for unsupervised knowledge transfer between any two layers of neural networks was proposed. The proposed method samples the geometry of the feature space, as induced by the donor model, and then trains the receiver model to mimic this geometry using similarity-induced embeddings. In contrast to existing approaches, the proposed method does not require having access to the original training dataset or to the weights of the donor model and it supports directly transferring the knowledge between neural networks without using dimensionality reduction to match the dimensionality of the layers. The effectiveness of the proposed method was demonstrated through extensive experiments.

Acknowledgment

This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 731667 (MULTIDRONE). This publication reflects the authors' views only. The European Commission is not responsible for any use that may be made of the information it contains.

References

[1] L. Shao, D. Wu, and X. Li, "Learning deep and wide: A spectral method for learning deep networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 12, pp. 2303–2308, 2014.

[2] Y. Yuan, L. Mou, and X. Lu, "Scene recognition by manifold regularized deep learning architecture," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 10, pp. 2222–2233, 2015.

[3] C. Bucilu, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, 2006, pp. 535–541.

[4] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[5] T. Chen, I. Goodfellow, and J. Shlens, "Net2net: Accelerating learning via knowledge transfer," *arXiv preprint arXiv:1511.05641*, 2015.

[6] N. Passalis and A. Tefas, "Dimensionality reduction using similarity-induced embeddings," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2017.

[7] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," *arXiv preprint arXiv:1412.6550*, 2014.

[8] W. Chan, N. R. Ke, and I. Lane, "Transferring knowledge from a rnn to a dnn," *arXiv preprint arXiv:1504.01483*, 2015.

[9] Z. Tang, D. Wang, and Z. Zhang, "Recurrent neural network training with dark knowledge transfer," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 5900–5904.

[10] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, "Simultaneous deep transfer across domains and tasks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4068–4076.

[11] Z. Tang, D. Wang, Y. Pan, and Z. Zhang, "Knowledge transfer pre-training," *arXiv preprint arXiv:1506.02256*, 2015.

[12] J. Davis and P. Domingos, "Deep transfer via second-order markov logic," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009, pp. 217–224.

[13] X. Shu, G.-J. Qi, J. Tang, and J. Wang, "Weakly-shared deep transfer networks for heterogeneous-domain knowledge propagation," in *Proceedings of the 23rd ACM International Conference on Multimedia*, 2015, pp. 35–44.

[14] J. Hu, J. Lu, and Y.-P. Tan, "Deep transfer metric learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 325–333.

[15] X. Zhang, F. X. Yu, S.-F. Chang, and S. Wang, "Deep transfer network: Unsupervised domain adaptation," *arXiv preprint arXiv:1503.00591*, 2015.

[16] Y. LeCun, C. Cortes, and C. J. Burges, "The mnist database of handwritten digits," 1998.

[17] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[18] M. Koestinger, P. Wohlhart, P. M. Roth, and H. Bischof, "Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization," in *IEEE International Workshop on Benchmarking Facial Image Analysis Technologies*, 2011.

[19] "Tiny ImageNet dataset," https://tiny-imagenet.herokuapp.com, [Online; accessed 10-Jan-2018].

[20] "notMNIST dataset," http://yaroslavvb.com/upload/notMNIST, [Online; accessed 1-March-2017].

[21] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[25] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and $<$ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

Nikolaos Passalis obtained his B.Sc. in informatics in 2013 and his M.Sc. in information systems in 2015 from Aristotle University of Thessaloniki, Greece. He is currently pursuing his Ph.D. studies in the Artificial Intelligence & Information Analysis Laboratory in the Department of Informatics at the University of Thessaloniki. His research interests include deep learning, information retrieval and computational intelligence.

Anastasios Tefas received the B.Sc. in informatics in 1997 and the Ph.D. degree in informatics in 2002, both from the Aristotle University of Thessaloniki, Greece. Since 2017 he has been an Associate Professor at the Department of Informatics, Aristotle University of Thessaloniki. From 2008 to 2017, he was a Lecturer, Assistant Professor at the same University. From 2006 to 2008, he was an Assistant Professor at the Department of Information Management, Technological Institute of Kavala. From 2003 to 2004, he was a temporary lecturer in the Department of Informatics, University of Thessaloniki. From 1997 to 2002, he was a researcher and teaching assistant in the Department of Informatics, University of Thessaloniki. Dr. Tefas participated in 12 research projects financed by national and European funds. He has co-authored 85 journal papers, 193 papers in international conferences and contributed 8 chapters to edited books in his area of expertise. Over 3730 citations have been recorded to his publications and his H-index is 32 according to Google scholar. His current research interests include computational intelligence, deep learning, pattern recognition, statistical machine learning, digital signal and image analysis and retrieval and computer vision.