[RE] Zero-Shot Knowledge Transfer via Adversarial Belief Matching

Alexandros Ferles[*]
ferles@kth.se

Alexander Nöu[*]
anou@kth.se

Leonidas Valavanis[*]
leoval@kth.se

Abstract

We reproduce the work in *Zero-shot Knowledge Transfer via Adversarial Belief Matching*, which describes a novel approach for knowledge transfer. A teacher network trained on real samples distills knowledge to a student network that is trained solely on pseudo data extracted from a generator network, with the student trying to mimic the teacher's outputs on these data. To this end, we additionally reproduce Wide Residual Networks which are used as the main framework for both teacher and student networks trained from scratch on CIFAR10 and SVHN. We compare the results of proposed method with a few-shot knowledge distillation attention transfer setting implemented and trained also from scratch. We suggest an approach for further exploitation of the learnt mechanics of the generator network in the zero-shot setting, which operates on top of the main method, and briefly discuss the benefits and drawbacks of this approach. Our code can be found publicly available in https://github.com/AlexandrosFerles/NIPS_2019_Reproducibilty_Challenge_Zero-shot_Knowledge_Transfer_via_Adversarial_Belief_Matching

1 Introduction

Knowledge distillation is a model compression technique which attempts to transfer the knowledge of large cumbersome models to smaller models. Many recent successful deep networks are extremely large and contain millions of parameters, which limits their usage to machines with more powerful hardware. For such networks to be available to a wider range of devices, model compression techniques are vital.

For this project, we reproduce the paper *Zero-shot Knowledge Transfer via Adversarial Belief Matching* [1], where the authors present a method for distilling the knowledge of a larger pre-trained network to a smaller one, without the use of real data. Our work consists of a full re-implementation and reproduction of all the methods and experiments in the paper, including re-training the Wide Residual Network[2] teacher network scratches on CIFAR10 and SVHN, reproducing the few-shot knowledge distillation via attention transfer of [3], as well as providing a modification of the main method in an attempt to yield better zero-shot knowledge transfer results. We analyze our findings and fully present our results, and discuss the reproducibility process of the paper and discrepancies when comparing to source code.

The rest of this paper is structured as follows: we first describe the network architectures and methods which we reproduced as part of our work in in section 2, and discuss implementation-specific details in section 3. Our complete experimental setting is presented in section 4, followed by the results that we derived from each method at section 5. Our conclusions and points for future improvements of the main methods, are presented in section 6. A comprehensive list of all our individual experiments and results which were carried out through this work can be found in the Appendix section.

[*] Equal contribution

2 Methods

2.1 Wide Residual Networks

Wide Residual Networks (WRNs) were originally proposed in [2] and are used as the main framework for both the teacher and student network in the few-shot knowledge distillation setting of [3] and zero-shot knowledge transfer setting of [1]. The main motivation of WRNs is to provide a network with similar performance to much deeper neural networks by taking advantage of less yet wider residual layers. WRNs are uniquely defined by two hyperparameters: the depth n of the network and the width factor w of each layer.

WRNs consist of a single convolutional layer, followed by 3 blocks of convolutional layers that extract features which are subsampled by a *global average pooling* layer before being fed to a linear layer to generate class predictions. The amount of convolutional layers at each block is the same, and is defined by the factor n of the network. Additionally, each convolutional layer which lies inside the blocks of WRNs, learns a residual function[4] on its input. The initial convolutional layer on all WRN versions is the same and performs a convolution operation that outputs 16 feature maps. In their simpler form ($n = 16$, $k = 1$) WRNs use 16, 32 and 64 output feature maps at each block in respect. Wider version multiply each of these values with k to define the amount of feature maps that will be used in each block.

At each individual block, the first convolution operation is responsible to subsample its input and increase the number of feature maps, when necessary. Finally, the operations of batch normalisation and ReLU activation are applied in reverse order compared to most deep convolutional networks, as in WRNs each batch normalisation layer precedes the non-linearity activation function.

2.2 Knowledge Distillation and Attention Transfer

The zero-shot method is evaluated through comparison with a few-shot knowledge distillation method proposed in [5]. A student network matches the outputs of a pre-trained teacher network by feeding real data to the teacher and using the output probabilities as targets for training the student. The original paper uses cross entropy loss to train the student with softened teacher probability outputs

$$q_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}, \tag{1}$$

where temperature T yields a softer probability distribution of classes and (1) corresponds to standard softmax activation of the teacher outputs z_i when $T = 1$. To make use of the true labels of the data, a weighted combination of cross entropy losses with labels and teacher outputs as targets serve as objective for training the student. In the experiments of [1], the cross entropy with teacher outputs is switched out for the Kullback-Leibler divergence. Moreover, the baseline model is augmented by adding an attention transfer loss [3]:

$$\beta \sum_{l}^{N_L} \left|\left| \frac{f(A_l^{(t)})}{\left|\left|f(A_l^{(t)})\right|\right|_2} - \frac{f(A_l^{(s)})}{\left|\left|f(A_l^{(s)})\right|\right|_2} \right|\right|_2 . \tag{2}$$

The additional loss takes a subset of activation blocks A_l and computes the squared mean over channels in order to get spatial attention maps $f(A_l)$. By adding (2) to the objective, the student is encouraged to match the spatial attention maps of the teacher.

As in the zero-shot setting that follows, WRNs will be used for both the teacher and student network. They can easily be integrated in attention-based knowledge transfer methods since we can make use of the output feature maps of each block as a point of comparison, while the spatial resolution of the output of each block is the same regardless of the WRN version. Hence there is no need for interpolation. Additionally, since we aggregate over the filter dimension in order to create the spatial attention maps of each output, we can effectively compare teacher and student WRNs of different widths without extra operations on this dimension, such as a linear mapping to the same filter dimension.

To follow the notation of [3] and [1] for the rest of this paper we refer to this method as KD-AT.

2.3 Zero-Shot Knowledge Transfer

The proposed idea of zero-shot knowledge transfer from teacher to student network is to introduce a generator network, and train the student and the generator adversarially. Following the notations of [1], we let $T(x)$, $S(x;\theta)$ and $G(z;\phi)$ be pretrained teacher network, student network and generator, where the weights θ and ϕ parameterize their respective networks that are to be trained. In order to train the student to match the teacher without real data, we sample noise $z \sim \mathcal{N}(0, I)$ and let $G(z)$ generate fake samples x_p. Gradient updates are alternated between student and generator to optimize the Kullback-Leibler divergence:

$$D_{KL}(T(x_p) \mid\mid S(x_p)) = \sum_i t_p^{(i)} \log \left(\frac{t_p^{(i)}}{s_p^{(i)}} \right), \tag{3}$$

where t_p, s_p are the teacher and student output probabilities given pseudo-data, and i denotes the class. The student minimizes (3) to force it to match the output probabilities of the teacher, which the authors call "belief matching". For the generator, the objective is instead maximized so that it learns to produce samples that make it more difficult for the student to match the teacher. The adversarial belief matching is balanced through an appropriate choice in numbers of gradient steps n_G, n_S when alternating the training. In addition, the authors experiment with extra loss functions. The attention transfer term (2) used for the baseline is also applied to the zero-shot model for the main experiments of the paper.

Thus, zero-shot knowledge transfer performs based on the samples drawn from the generator network, and distills knowledge to a student network by matching the outputs of the teacher much like the knowledge distillation approach in [5]. The advantage, however, becomes clear compared to the case when only a few training samples are available: Instead of learning to match the teacher on the same limited number of samples at each iteration, the generator is trained alongside the student and will continue to generate challenging pseudo-data to further close the gap between student and teacher.

2.4 Modification of the Zero-Shot Method

In the training setting of the main method, for each iteration the generator creates a single batch of samples (when $n_g = 1$) on which we then train the student network n_s times to match the teacher's feature maps and outputs on this sole batch. The motivation for using only one batch is clear and justified, since we first force the samples in a position of the sample space which makes it difficult for the student to learn, which requires multiple student updates to balance the training. However, we propose a slight modification on the zero-shot training setting. Instead of using the same sample that the generator was updated on, we use the updated generator to provide us with n_s different batches (keeping n_G the same as before), each of them used only once to update the student based on the teacher's outputs on them. This way, we intend to create a more diverse pseudo-training dataset which could provide an improved training setting for the student network.

2.5 Belief Matching

In order to measure the student network's degree of belief matching (network's ability to match output probabilities of teacher) with its teacher, the following procedure is executed: Test samples are progressively changed in the direction of the student's decision boundary until they resemble input data of another class. As the student's confidence in a sample belonging to the other class increases, we monitor the predictions of teacher as well. Ideally, we would expect the teacher to follow closely the behaviour of the student. We thus iterate over a number of test samples whose predicted labels are the same for student and teacher. Then, iterating over all possible classes that are not the predicted one, we take K steps of gradient updates on the sample to alter it towards the "fake" class j with learning rate ξ. We get the gradient by feeding the sample to the student and computing the cross entropy with class j as target. In each step we let both student and teacher predict the progressively altered sample, and store their respective probability p_j of the sample belonging to class j. Finally, the mean over fake classes and test data size results in a transition curve of p_j over K steps. Mean

Transition Error is introduced to quantify this matching capability, and computed through the absolute difference $|p_j^s - p_j^t|$ between networks and taking the mean over K steps, fake classes and sample size.

3 Implementation

3.1 TensorFlow implementation

In view of the author's use of PyTorch as framework for implementations, we also attempted to re-implement the WRNs used in the main and few-shot papers by using TensorFlow. Achieving the results of the papers, however, proved to be rather difficult because of the numerous implementation details that change when transitioning from PyTorch to TensorFlow. Small details such as the weight initialization methods of various layers affect performance. Other discrepancies between the frameworks can be found in parameter choices for the built in dataset transformations, batch normalization layers, convolutional layers and optimizers. For example, momentum for updating running statistics in batch normalization turns out to be oppositely defined in TensorFlow, and TensorFlow offers no easy way to apply weight decay to the SGD optimizer directly. Therefore we must further study components of the frameworks and the paper's source code in detail for accurate reproduction. In the end, the baseline architectures performed somewhat worse, which is why we decided to proceed with PyTorch for further experiments. We also provide our TensorFlow code for GitHub for full disclosure.

3.2 Discussion on Reproducibility Issues

For all of the methods used to derive the results of this paper, we used the PyTorch framework to train our deep networks along with the external components such as Adversarial Belief Matching. We first designed each method on our own, and then consulted the official codes of the zero-shot and few-shot knowledge transfer to find hyperparameter values (listed thoroughly in the Appendix section) and fine-tune our networks. In detail, we had to integrate the following settings in our work, which were not mentioned in the paper but implemented in the official repository of the authors:

- To our knowledge, there is no mention about weight initialization in [2] or [3] from the authors of Wide ResNets. We thus used the weight initialization presented on GitHub.
- We initially treated the hyperparameters of Temperature T and α value on knowledge distillation between the teacher and student network as presented in [5], and then changed it to the values used by the authors on [3].
- In attention transfer, the authors in [3] suggest that the best way to extract the spatial attention map would be to use the sum of the square of each individual pixel per channel, but the authors of [1] use the squared mean. Furthermore, the distance between student and teacher maps is quantified by taking the squared mean over batch and spatial size, as opposed to Euclidean distance which they state in their paper.
- In [3] and [5], cross entropy is used for the student's loss term with teacher outputs as targets. However, in both the few-shot KD and zero-shot settings of [1] teacher and student are compared with the use of KL divergence between the softmax activations of the former and the log-softmax of the latter (KL for the zero-shot model is stated in the paper).
- There is no description of the Generator network in [1] apart from "We use a generic generator with only three convolutional layers, and our input noise z has 100 dimensions". Thus, we consulted the official code for more details in order to design this network.
- In the zero-shot method of [1] the paper does not mention that weight clipping is performed on both the student and generator networks. We proceeded with integrating weight clipping to our training too.

4 Experiments

4.1 Data and Preprocessing

The network is evaluated on two commonly used data sets, *CIFAR-10* and *SVHN*, that include 60000 and approximately 100000 32x32 images respectively, with the majority of these images used at a training set. The only preprocessing method applied on SVHN is mean/std normalization. On the other hand, we perform a few methods of data augmentation on CIFAR-10 in addition to normalization, namely reflect mode image padding, random cropping and random horizontal flipping.

4.2 Training WRN Scratches

The batch size on both datasets is equal to 128, and in order to match the update steps claimed on [1], we trained CIFAR10 for 200 epochs and SVHN for 100 epochs respectively. For both datasets, we apply a Stochastic Gradient Descent (SGD) optimizer with Nesterov momentum (equal to 0.9) and a weight decay of $5 * 10^{-4}$. The initial learning is equal to 0.1 and divided by 5 when 30%, 60% and 80% of the update steps have been completed. Most of the steps were directly motivated from [1], while we also consulted [3] and [2] when some settings were not clear to us. We apply three seeds on each training, namely 0, 1 and 2, and apart from our own method described in 2.4 we use the same 3 seeds for the rest of this work. As in [1], we train 4 variants of WRN, namely WRN-16-1, WRN-16-2, WRN-40-1 and WRN-40-2.

We also trained few-shot scratches of WRN-16-1 on M samples per class (M drawn again from {10, 25, 50 , 75, 100}) on each dataset under the same configurations, to generate the 'No Teacher' models mentioned in [1]. In order to train for the same number of update steps, we scale the number of original epochs based on each training dataset size and the value of M, by the following formula:

$$epochs' = \frac{Dataset\_Size}{10 * M} * epochs \tag{4}$$

Lastly, we evaluate WRN-16-1 directly on each test set to mimic the 'No Teacher' model with M=0. Since this is exact setting of KD-AT with M=0, we only train this setting once per seed for both cases.

4.3 Few-Shot Knowledge Attention Transfer

For few-shot knowledge distillation with attention transfer, we train WRN-16-1 under the same hyperparameter settings for each dataset and values of M drawn from {10, 25, 50 , 75, 100} for WRN-16-1 for both CIFAR and SVHN, and M=5000 for knowledge distillation when trained on full data. Additionally, we combine all the possible teacher-student pairs of the 4 variants of WRN (listed in table 1) to train the KD-AT setting for M=200 on CIFAR10. Formula (4) is once again used to define the number of training epochs for each dataset and value of M.

4.4 Zero-Shot and Modified Zero-Shot Training

The zero-Shot training setting relies on training with fake samples, so we do not need to scale the number of epochs. Instead, for both CIFAR10 and SVHN we train for 80000 iterations, sample a pseudo-batch and update the generator once per iteration ($n_g = 1$) and then update the student $n_s = 10$ times per iteration. For the modified zero-shot model, the generator produces a new batch for each student update. We switch to Adam optimizer[6] with cosine annealing[7], with an initial learning rate of $2 * 10^{-3}$ for the student network and $1 * 10^{-3}$ for the generator. Noisy inputs are sampled from a standard normal distribution with 100 dimensions, and fed to the generator which extracts pseudo batches of size 128*3*32*32, like the input batches of WRNs when trained on real data. The models are in addition fine-tuned for few-shot distillation comparisons using the KD-AT procedure for a further 200 epochs. While the value of M is not clearly stated on [1] for the SVHN data, we perform a KD-ATT training with M=200 to match the case with the CIFAR data.

4.5 Measuring Belief Matching

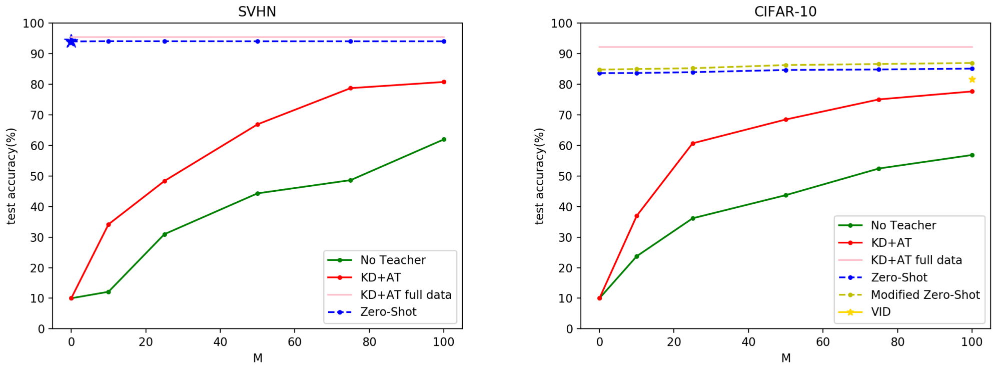
For the belief matching experiment, we make use of a WRN-40-2 teacher and two WRN-16-1 students, one trained from the KD-AT setting and another one trained from the zero-shot setting. The paper does not state which M is used for KD-AT. Hence, we choose $M = 200$ for fair comparison as it has similar test accuracy to the zero-shot model. In order to compute the probability transition curves described in 2.5, the process is guided from a learning rate ξ equal to 1, and 100 update steps are performed per sample and fake label. For each of CIFAR10 and SVHN, we use 1000 test set samples, and average over the extracted probability transition curves to display our results. We also compute the Mean Transition Error (MTE) between the teacher and each of the students on each dataset as in [3] via the formula:

$$\frac{1}{N_{samples}}\sum_{n=1}^{N_{samples}}\frac{1}{C-1}\sum_{n=1}^{C-1}\frac{1}{K}|p_{student}-p_{teacher}| \tag{5}$$

where $N_{samples}$ represents the 1000 samples from each dataset, C is the number of different classes (equal to 10 for both CIFAR10 and SVHN), K represents the 100 updates steps, $p_{student}$ and $p_{teacher}$ are the probability estimations of the student and teacher for each of the K update steps on C-1 fake labels and $N_{samples}$ samples.

5 Results

We first reproduce the zero-shot and few-shot experiments for the teacher and student architectures WRN-40-2 and WRN-16-1 on SVHN and CIFAR-10. The results are presented in Figure 1, which shows test accuracies of the baseline model KD-AT trained with M samples or all data, as well as the zero-shot model and a student trained from scratch with M samples. The test accuracies are the means over three trials. In addition, the results include the performance of Variational Information Distillation (VID) of [8]. We can see in Figure 1 that the results of the paper are reproduced, where the zero-shot model outperforms KD-AT and VID trained with $M = 100$ samples, and almost reaches the accuracy of KD-AT with $M = 5000$ on SVHN.



(a)

(b)

Figure 1: Performance for different algorithms using SVHN (a) and CIFAR-10 (b) datasets. Variational information distillation (VID)[8] has a single value for the CIFAR-10 dataset using M=100 samples per class.

Table 1 shows reproduced results of the experiment investigating architecture dependence on CIFAR-10. The mean test accuracies over three trials are close to the results of the paper, with smaller discrepancies. Similar to their results, we can also notice that zero-shot distilling WRN-40-2 to WRN-16-2 performs better than distilling to WRN-40-1 (somewhat closer than the paper), suggesting that deeper student networks with similar number of parameters not necessarily perform better. The opposite can be seen for KD-AT, with the deeper student network performing best (but with larger standard deviations than the paper). Moreover, we include results of our modified zero-shot algorithm, which show improved performance for all network architectures. Training our modified algorithm requires multiple generated batches per iteration, and results in higher complexity in terms of speed. However, it converges to a similar or higher accuracy in fewer iterations of the training process, making it run in a similar time or sometimes faster than the original zero-shot algorithm. Due to the complexity of the task, we did not have enough resources to further evaluate the performance of the algorithm.

Teacher (# params)	Student (# params)	Teacher scratch	Student scratch	KD+AT M = 200	Zero-Shot	Modified Zero-Shot
WRN-16-2 (691K)	WRN-16-1 (175K)	94.21 ±0.03	91.38 ±0.33	85.55 ±0.25	81.25 ±0.86	82.82 ±1.09
WRN-40-1 (563K)	WRN-16-1 (175K)	93.83 ±0.22	91.38 ±0.33	83.64 ±0.22	79.90 ±1.82	82.61 ±2
WRN-40-2 (2.243M)	WRN-16-1 (175K)	95.16 ±0.04	91.38 ±0.33	82.85 ±0.95	83.63 ±0.15	84.78 ±0.5
WRN-40-1 (563K)	WRN-16-2 (691K)	93.83 ±0.22	94.21 ±0.03	87.25 ±0.18	87.71 ±0.71	89.27 ±0.6
WRN-40-2 (2.243M)	WRN-16-2 (691K)	95.16 ±0.04	94.21 ±0.03	87.27 ±0.69	89.31 ±0.14	91.12 ±0.32
WRN-40-2 (2.243M)	WRN-40-1 (563K)	95.16 ±0.04	93.83 ±0.22	88.41 ±0.64	87.46 ±0.33	90.27 ±0.22

Table 1: Zero-shot and modified zero-shot results versus few-shot attention transfer (KD+ATT) using WRN for CIFAR-10 and SVHN. Results display mean and standard deviation over 3 seeds.

Overall, we observe that even in our reproducibility work, we get slightly better results on the same settings as [1]. Since on this part we tried to stay as close as possible to the methods that were reported, we mostly attribute the improvements to the data augmentation that we applied on CIFAR when optimizing the scratch networks, which we also retained in all of our settings. Additionally, we observed that the modified zero-shot setting brings improvements even close to 3 percentage points for some cases. Our intuition is that this can be attributed to the greater diversity of samples drawn from the generator, which was our main motivation for introducing this method. The accuracy of both zero-shot settings can slightly increase if we switch to few-shot training by taking extra update steps on the student on a few real samples, however this increase stays limited (at a few cases there was no improvement at all) which hints us that the majority of the necessary features have already been learned by the student when trained on the zero-shot settings.

Samples drawn from different generator networks at different stages of their training can be seen in the following figure. Through visual inspection, we observe that starting from random noise (as expected), features start to grow dependencies and form patterns that are useful for network training and can serve as a substitute of real data, when the latter are not available.

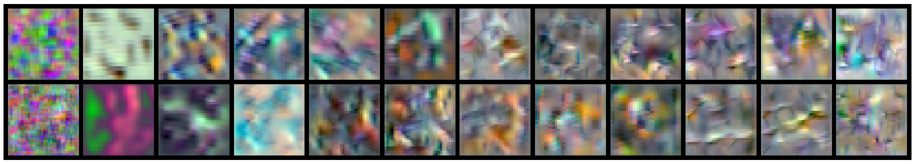


Figure 2: Pseudo images sampled from generators of different seed, hyperparameters and Teacher-Student pairs at different times during training. As the training develops (from left to right) the images evolve from diverse, random features to shaped patterns.

We finally measure the belief matching between teacher and student in both the zero-shot and few-shot settings for both datasets. Figure 3 depicts the reproduced transition curves for all four cases, and Table 2 shows MTE (equation 5). The performance of the zero-shot model is very similar to the paper, but the transition error of KD-AT is higher. We observe the same pattern as the authors, that similarity in predictions between student and teacher as samples are altered is much worse for KD-AT, despite having comparable test accuracy to the zero-shot model. This is surprising since the procedure is using real data, which KD-AT uses for distillation. We provide a possible explanation for this: The process of manipulating samples towards the student decision boundary might result in images outside the space of real data. Examples of images after K update steps towards the student decision boundary of other classes can be found in the appendix. The images look like noisy versions of the original class, but are now predicted as another class with almost full certainty by the student. For KD-AT, the student matches the teacher and true labels solely on real samples, whereas the zero-shot student is trained on pseudo-data which is not limited to this space, as is shown in Figure 2. A toy experiment is also conducted in [1], demonstrating how the generator produces samples that follow the decision boundary of the teacher in order to make it more difficult for the student, which could explain the high degree of belief matching in our experiments.

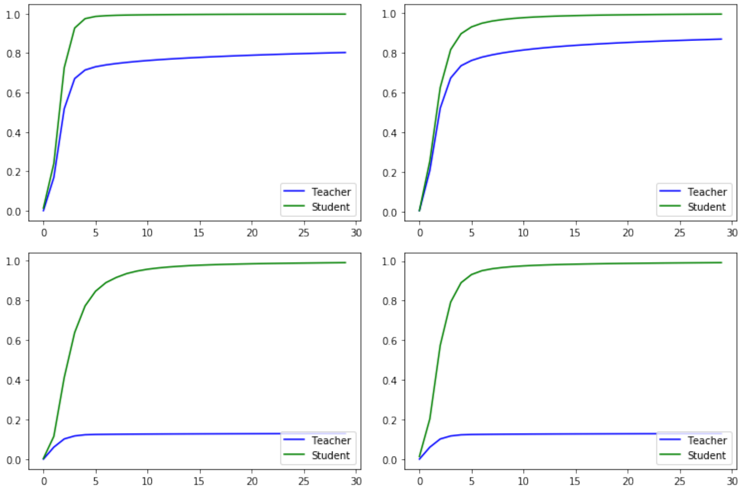


Figure 3: Transition curves of teacher and student network when samples from the test sets of CIFAR10 and SVHN are manipulated to change their labels. **Top row**:Zero-Shot training on CIFAR (left) and SVHN (right) **Bottom row**:Corresponding results for few-shot attention transfer on each dataset.

	Zero Shot	**KD+AT**
SVHN	0.11	0.83
CIFAR-10	0.18	0.81

Table 2: Mean Transition errors (MTE) for SVHN and CIFAR-10

6 Conclusions and Future Work

In this project, we reproduced the zero-shot knowledge transfer proposed in [1]. By firstly training a generator to produce images on which the student fails to match its teacher and then the student to mimic the decisions of the teacher with these pseudo data, we end up with a setting that performs similar or better than few-shot algorithms in simple datasets such as SVHN and more diverse datasets such as CIFAR-10. Moreover, we modified the training setting and generated new images at each student gradient update instead of once in the beginning of the iteration. Consequently, the dataset is more diverse for the student to learn and the algorithm converges to the same accuracy in less iterations.

The initial work along with the modifications we proposed leave room for further exploration and analysis. For example, the generator that provides adversarial samples for distilling knowledge when no real samples are available, at the moment comprises of a few convolution, batch normalization and non-linearity activation operations. We expect a more thoroughly designed generator to provide us with adversarial features of better quality in terms of optimizing a student network. In fact, our observations from our own modification encourage us to explore several further settings for this part of the zero-shot training. In-depth analysis of generated pseudo data and its diversity could also be performed, so that the resulting modified zero-shot model can provide additional insight to what effect sampling multiple batches has on the student network. Another possible research direction, would be to further explore the usability of the fact that intermediate feature maps are also optimized through the attention transfer loss. In [9], visual attention is applied to the VGG network[10] by scaling middle and coarse layer feature maps in combination with the output feature maps to improve its performance compared to its baseline version. Thus, we may be able to use a method similar to this work, to further increase the efficiency of the student network.

References

[1] Paul Micaelli and Amos J. Storkey. Zero-shot knowledge transfer via adversarial belief matching. *CoRR*, abs/1905.09768, Accepted in Conference of Neural Information and Processing Systems, 2019. URL http://arxiv.org/abs/1905.09768.

[2] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press, September 2016. ISBN 1-901725-59-6. doi: 10.5244/C.30.87. URL https://dx.doi.org/10.5244/C.30.87.

[3] Sergey Zagoruyko and Nikos Komodakis. 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*, 2017.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. doi: 10.1109/CVPR.2016.90.

[5] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[6] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[7] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

[8] Sungsoo Ahn, Shell Xu Hu, Andreas C. Damianou, Neil D. Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *CVPR*, 2019.

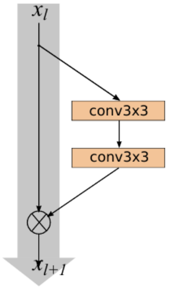
[9] Saumya Jetley, Nicholas A. Lord, Namhoon Lee, and Philip H. S. Torr. Learn to pay attention. *ArXiv*, abs/1804.02391, 2018.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. URL http://arxiv.org/abs/1409.1556.

A Wide Res Net architecture

The architecture of Wide ResNet is summed on the following figure:

group name	output size	block type = $B(3,3)$
conv1	32×32	[3×3, 16]
conv2	32×32	$\begin{bmatrix}3\times3, 16\times k \\ 3\times3, 16\times k\end{bmatrix}\times N$
conv3	16×16	$\begin{bmatrix}3\times3, 32\times k \\ 3\times3, 32\times k\end{bmatrix}\times N$
conv4	8×8	$\begin{bmatrix}3\times3, 64\times k \\ 3\times3, 64\times k\end{bmatrix}\times N$
avg-pool	1×1	$[8\times8]$



General structure of a Wide Residual Network (left) and form of a single residual convolutional layer at each block, as presented in [3]. The factor N which defines how many convolutional layers will be used at each block is not to be confused with the depth n of the network, and is directly depended on n via the formula $N = (n-4)\ div\ 6$.

B Full Experiments

B.1 Training scratches of Wide ResNets

In order to use Wide ResNets of different depth and width as teacher networks for the Few-Shot Attention Knowledge Distillation and the Zero-Shot Knowledge Transfer, we trained 4 variants of Wide ResNet from scratch. The results on CIFAR10 are shown in Table 3 below.

Model	**Seed 0**	**Seed 1**	**Seed 2**
WRN-16-1	90.97	91.41	91.78
WRN-16-2	94.21	94.27	94.18
WRN-40-1	93.52	94.04	93.94
WRN-40-2	95.14	95.12	95.23

Table 3: Wide ResNet scratches performance on *CIFAR-10*

The results of training teachers on SVHN are shown in Table 4 below.

Model	**Seed 0**	**Seed 1**	**Seed 2**
WRN-16-1	95.52	95.43	95.47
WRN-16-2	96.17	96.09	96.03
WRN-40-1	96.07	96.14	96.19
WRN-40-2	96.14	96.13	96.37

Table 4: Wide ResNet scratches performance on *SVHN*

B.2 Training Wide ResNet 16-1 with no Teacher

We also trained WRN-16-1 from scratch on small subsets of M images per class on CIFAR10 and SVHN and without the use of a teacher network to assist in the learning process. We firstly show the results on CIFAR10 in Table 5 below.

M	Seed 0	Seed 1	Seed 2
10	23.7	21.68	25.86
25	34.4	38	36.07
50	41.69	44.2	45.27
75	54.45	51.89	50.99
100	57.02	56.87	56.69

Table 5: Wide ResNet 16-1 few-shot training on *CIFAR-10* with no assistance from a teacher network

The results on SVHN are the following in Table 6:

M	Seed 0	Seed 1	Seed 2
10	11.97	12.67	11.73
25	31.83	34.21	26.82
50	44.08	45.93	42.93
75	50.07	41.88	53.96
100	56.71	69.58	59.56
200	87.65	87.92	87.18

Table 6: Wide ResNet 16-1 few-shot training on *SVHN* with no assistance from a teacher network

B.3 Few-Shot Knowledge Distillation with Attention Transfer (KD-AT)

Few-Shot Knowledge Distillation with Attention Transfer is trained using different pairs of Teacher-Student and different values of M for CIFAR-10 and SVHN datasets. We firstly show the results on CIFAR10 using WRN-40-2 for the Teacher and WRN-16-1 for the Teacher, for different values of M in Table 7.

M	Seed 0	Seed 1	Seed 2
10	39.08	35.33	36.49
25	60.05	58.94	63.05
50	70.9	65.83	68.68
75	73.84	74.29	77
100	76.67	76.72	79.57
5000	92.15	92.25	92.17

Table 7: Few-shot training on *CIFAR-10* with Attention Transfer using WRN-40-2 for the Teacher and WRN-16-1 for the Student, for different values of M

The results on SVHN using WRN-40-2 for the Teacher and WRN-16-1 for the Student, for different values of M are shown in Table 8.

M	Seed 0	Seed 1	Seed 2
10	37.35	31.32	33.88
25	48.71	48.89	47.44
50	68.84	65.33	66.48
75	78.51	78.4	79.28
100	81.18	79.63	81.45
5000	95.19	95.44	95.72

Table 8: Few-shot training on *SVHN* with Attention Transfer using WRN-40-2 for the Teacher and WRN-16-1 for the Student, for different values of M

The results for different pairs of Teacher-Student for M=200 is shown in Table 9.

Teacher	Student	Seed 0	Seed 1	Seed 2
WRN-16-2	WRN-16-1	85.51	85.26	85.89
WRN-40-1	WRN-16-1	83.9	83.35	83.67
WRN-40-1	WRN-16-2	87.52	87.14	87.13
WRN-40-2	WRN-16-1	82.41	81.97	84.18
WRN-40-2	WRN-16-2	87.15	86.49	88.18
WRN-40-2	WRN-40-1	88.18	87.77	89.29

Table 9: Teacher-Student Wide ResNets few-shot training on CIFAR-10 with Attention Transfer, for *M = 200*

B.4 Zero-Shot Knowledge Transfer

We trained the zero-show Knowledge transfer algorithm for various pairs of Teacher Student for CIFAR-10 and SVHN. In Table 10 the results for CIFAR-10 is shown for various seeds and Teacher Student pairs and in Table 11 the experiment for SVHN is shown.

Teacher	Student	Seed 0	Seed 1	Seed 2
WRN-16-2	WRN-16-1	80.59	80.7	82.48
WRN-40-1	WRN-16-1	77.4	80.61	81.7
WRN-40-1	WRN-16-2	88.71	87.34	87.08
WRN-40-2	WRN-16-1	83.73	83.76	83.42
WRN-40-2	WRN-16-2	89.13	89.48	89.32
WRN-40-2	WRN-40-1	87.94	87.28	87.18

Table 10: Teacher-Student Wide ResNets zero-shot training on *CIFAR-10*

Teacher	Student	Seed 0	Seed 1	Seed 2
WRN-40-2	WRN-16-1	94.21	93.85	93.94

Table 11: Teacher-Student Wide ResNets zero-shot training on *SVHN*

B.5 Zero-Shot Knowledge Transfer with modified generator

Table 12 shows the results for the modified zero-shot we tried.

Teacher	Student	Seed 0	Seed 1	Seed 2
WRN-16-2	WRN-16-1	82.42	81.73	84.32
WRN-40-1	WRN-16-1	79.87	84.62	83.34
WRN-40-1	WRN-16-2	88.71	90.11	88.99
WRN-40-2	WRN-16-1	85.09	84.07	85.18
WRN-40-2	WRN-16-2	90.67	91.41	91.27
WRN-40-2	WRN-40-1	90.08	90.16	90.59

Table 12: Teacher-Student Wide ResNets zero-shot training on *CIFAR-10* with *modified generator*

B.6 Zero-Shot Knowledge Transfer with extra M real samples

Our results in CIFAR10 when extra samples are drawn from the generator, are presented in Table 15.

M	Seed 0	Seed 1	Seed 2
10	83.89	83.37	83.77
25	84.08	83.57	84.22
50	84.69	84.37	84.94
75	84.98	84.53	85.0
100	85.27	84.73	85.35

Table 13: Performance of Zero-Shot pre-trained student WRN-16-1 when few-shot knowledge distillation is performed from a teacher WRN-40-2 for a few epochs with M samples.

The same setting is repeated on the SVHN dataset in Table 14 with the following results:

M	Seed 0	Seed 1	Seed 2
10	94.29	93.9	94.0
25	94.26	93.97	93.98
50	94.26	93.94	93.97
75	94.27	93.95	93.94
100	94.24	93.97	93.94

Table 14: Performance of Zero-Shot pre-trained student WRN-16-1 when few-shot knowledge distillation is performed from a teacher WRN-40-2 for a few epochs with M samples.

B.7 Modified Zero-Shot Knowledge Transfer with extra M real samples

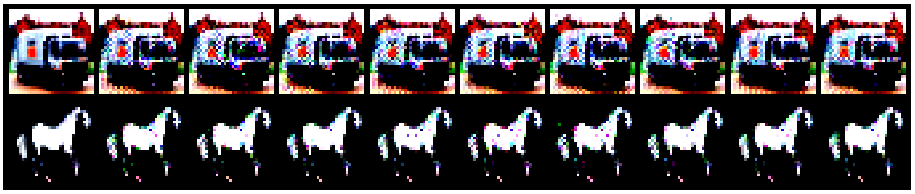
Our results in CIFAR10 when extra samples are drawn from the generator, are presented in Table 15.

M	Seed 0	Seed 1	Seed 2
10	85.09	84.54	85.31
25	85.09	85.21	85.43
50	86.37	86.18	86.29
75	86.77	86.4	86.67
100	87.2	86.74	86.96

Table 15: Performance of Zero-Shot pre-trained student WRN-16-1 when few-shot knowledge distillation is performed on top of our modified training method from a teacher WRN-40-2 for a few epochs with M samples.

C Samples for Measuring Belief Matching

The following figure shows images regarding the belief matching experiment conducted in section 4.5:



Two test samples from the measuring of belief matching. The figure shows the original images and the result of $K = 100$ altering steps towards each other class.