STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXV**, Number 2, 2020
DOI: 10.24193/subbi.2020.2.01

EXPERIMENTAL STUDY OF SOME PROPERTIES OF KNOWLEDGE DISTILLATION

ÁDÁM SZIJÁRTÓ, PÉTER LEHOTAY-KÉRY, AND ATTILA KISS

Abstract. For more complex classification problems it is inevitable that we use increasingly complex and cumbersome classifying models. However, often we do not have the space or processing power to deploy these models.

Knowledge distillation is an effective way to improve the accuracy of an otherwise smaller, simpler model using a more complex teacher network or ensemble of networks. This way we can have a classifier with an accuracy that is comparable to the accuracy of the teacher while small enough to deploy.

In this paper we evaluate certain features of this distilling method, while trying to improve its results. These experiments and examinations and the discovered properties may also help to further develop this operation.

1. Introduction

Knowledge distillation is a method to transfer the knowledge of an already trained neural network to another, possibly a smaller one. The benefit of this is we can achieve a higher accuracy for the student models as opposed to training it on their own.

This simple method can significantly boost the accuracy of a model in a way that could not be achieved with normal training methods or hard outputs. This distillation technique is especially useful when we have an accurate, but cumbersome neural network (or ensemble of networks); but do not have the resources to deploy it.

In this paper, we examine certain features of this method, such as transitivity and symmetry, by conducting experiments to prove or refute these

Received by the editors: 7 August 2020.

2010 *Mathematics Subject Classification.* 68T05, 68T30.

1998 *CR Categories and Descriptors.* I.2.6 [**Artificial Intelligence**]: Learning – *Connectionism and neural nets*;

Key words and phrases. Artificial Intelligence, Convolutional Neural Networks, Deep learning, Knowledge distillation, Neural networks.

properties. In addition, we evaluate how this method performs using an ensemble of student networks and how its parameters affect the results by further experiments.

We are going to evaluate the symmetry of knowledge distillation by distilling the knowledge back from a trained student network to an untrained teacher network. We are going to conclude that knowledge distillation is a symmetric operation if the new model performs better than the student model and its performance is close to the original model.

We are going to evaluate the transitivity of knowledge distillation by distilling the knowledge from the teacher model to a less complex middle model, and then distill it further to the original student model. We are going to conclude that knowledge distillation is a transitive operation if there is no significant difference between the accuracy of the middle model and the original student model.

2. Related works

In their 1998 paper titled "Neural network ensembles" [1], Lars Kai and Peter Salamon argue that building multiple classifier models and evaluating their results to a given classifying problem can vastly outperform a single model even if those models are significantly simpler and individually do not perform as well as the single model. They also found that cross validation could greatly reduce overfitting while training these models.

This idea was further elaborated in the 2000 paper "Ensemble methods in machine learning" [2] by T. G. Dietterich, who reviewed these methods and explained why ensembles could often perform better than any single classifier. Furthermore, the author reviewed some previous studies comparing ensemble methods and presented some new experiments.

Taking this as a basis in their 2015 paper "Distilling the knowledge in a neural network" [3] Hinton, Vinyals and Dean found that the "knowledge" from a trained complex model or even an ensemble of models could be distilled down into a much simpler model without compromising performance and accuracy. They argue that the training and deployment of a classifier are two completely different problems with different requirements. We should not use the same model, but use a cumbersome one for the training and – as the computational complexity is a huge factor for end users – we should use a distilled simpler model for deployment.

This idea has been further improved in the 2017 paper titled "A gift from knowledge transfer distillation: Fast optimization, network minimization and transfer learning" [4], which proposed a new solution: the knowledge from a pretrained deep neural network (DNN) is distilled and transferred to another

DNN. The method uses FSP (flow of solution procedure) matrix, representing the distilled knowledge from the teacher DNN.

In the "Distillation as a defense to adversarial perturbations against deep neural networks" [5] paper, the authors found that this method was also effective against adversarial attacks. They found that training a model, then distilling its knowledge to one that is structured the same way can significantly increase its robustness.

However, since then methods have been found for adversarial attacks against which this kind of distillation does not work. It has been elaborated in the paper titled "Towards Evaluating the Robustness of Neural Networks" [6] in 2017, where the authors introduced three new attack algorithms that were successful on both distilled and undistilled neural networks with 100% probability.

In addition to these, the idea of generating softened outputs with a trained classifier in order to enhance the performance of another one goes beyond neural networks. In their 2017 paper "Distilling a Neural Network Into a Soft Decision Tree" [7], Nicholas Frosst and Geoffrey Hinton argue that this method can be applied when distilling knowledge from a neural network to a decision tree.

"Residual Knowledge Distillation" [8] further distills the knowledge by introducing an assistant which learns residual errors. The experiments of the authors showed that their approach achieved appealing results on popular classification datasets.

The human visual system relies on temporal dependencies among frames from the visual input to conduct recognition. Based on this observation, "Tkd: Temporal knowledge distillation for active perception" [9] proposes the Temporal Knowledge Distillation framework, which distills the temporal knowledge from a neural network-based model over selected video frames to a light model. Results of the authors showed consistent improvement in accuracy-speed trade-offs for object detection, compared to other modern object recognition methods.

"Explaining Knowledge Distillation by Quantifying the Knowledge" [10] presents a method to qualify and analyze task-relevant and task-irrelevant visual concepts that are encoded in intermediate layers of a Deep Neural Network. Authors designed mathematical metrics to evaluate feature representations of the Deep Neural Network and diagnosed Deep Neural Networks as experiments.

"Learning an Evolutionary Embedding via Massive Knowledge Distillation" [11] proposes an Evolutionary Embedding Learning framework to learn a fast and accurate student network for open-set problems via Massive Knowledge

Distillation. Authors introduced a novel correlated embedding loss to match embedding spaces between the teacher and student network. EEL achieved better performance with other state-of-the-art methods for various large-scale open-set problems.

"Feature-map-level Online Adversarial Knowledge Distillation" [12] proposes an online knowledge distillation method that transfers the knowledge of the feature map using the adversarial training framework. Authors trained multiple networks simultaneously by employing discriminators to distinguish the feature map distributions of different networks. Furthermore, they proposed a novel cyclic learning scheme for training more than two networks together.

3. Background

3.1. **Convolutional Neural Networks.** For our experiments we used a CNN (Convolutional Neural Network)[13][14], which is a class of deep neural networks, a regularized multilayer perceptron. They are most often applied to analyze images, by learning filters independently from prior knowledge. CNNs consist of an input, an output and multiple hidden layers.

In neural networks, each neuron produces the output value by applying a function to the input values that come from the previous layer. Weights and biases determine this function and their iterative adjustments progress the learning.

In CNN, most of the hidden layers are convolutions, which are special linear operations. When data are passing through a convolutional layer, it becomes abstracted to a feature map.

CNNs may also include some pooling layers to reduce the dimensions of data. In our experiments we used max pooling[15][16]. Pooling combines the outputs of neurons in one layer into a single neuron in the next layer. Max pooling uses the maximum value as combination.

In order to reduce overfitting, we used Dropout [17][18] in each layer. Dropout means that at each training stage, nodes together with their edges are dropped out of the net with probability 1-p, so that a reduced network is left. Only this network is trained on the data at this stage. The removed nodes and edges are reinserted at the next stage.

3.2. **Knowledge distillation.** The knowledge distillation method uses a special activation function to produce "softened" probabilities, which are then used to train the student network, on which we also apply the previously mentioned activation function. This special function is a parameterized version of the widely used softmax[19] function, which is used to convert the last layer of the network into probabilities.

Softmax can be given in the following form[3]:

$$q_i = \frac{exp(\frac{z_i}{t})}{\sum_j exp(\frac{z_j}{t})}$$

where t is a parameter called temperature, which converts z_i logit value to q_i probability. For a standard softmax, t is normally set to 1. The higher we set this parameter, the softer the output probabilities are going to be, and this way we can preserve more features of the input than the teacher net learned, meaning that the student receives more information as opposed to using hard outputs.

The distilled model will be the smaller network we have trained on a transfer set, which is not the same dataset as the one we used to train the larger model. As loss function, cross entropy is used between the output of the distilled model and the output of the larger model.

4. Experiments

For the experiments we used the GTSRB (German Traffic Sign Recognition Benchmark) dataset[20]. The teacher net was a CNN (Convolutional Neural Network) with three layers, each with 128 nodes, using rectified linear activation functions.

With this model we managed to achieve an 0.9473 accuracy on the test set. This served as a baseline for our further experiments. As for the student, we used a dense neural network with one hidden layer with rectified linear activation function.

Training it normally with the hard outputs and traditional *softmax* output layer, we had an accuracy on the test set that is not higher than 0.1635. The results of the distillation process, in relation to the temperature parameter, can be seen in **Figure 1**.

Compared to the traditional training approach, we can clearly see a significant improvement in the graph. However, the temperature parameter does not seem to show much influence on the results if it is greater than 4. In fact the accuracy appears to be quite random between the range of 0.5 and 0.8. It will serve as a baseline in our further experimentation.

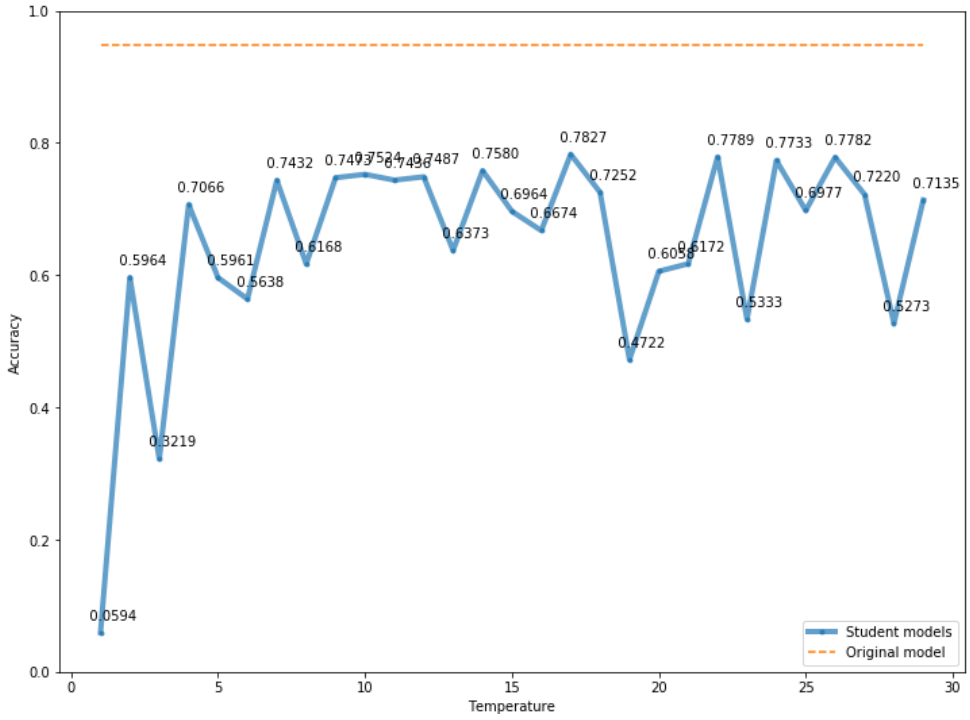

FIGURE 1. Results of the distillation

4.1. **Symmetry.** To test the symmetry of the method, we investigated if we could reverse the distillation process by taking a student model that had been trained with this technique, then we distilled its knowledge into the original (untrained) teacher model. For this experiment we took the best performing student network – which we received with temperature 17, and had an accuracy of 0.7827 – then used it to generate the softened outputs.

They were used to train the teacher model with the modified softmax output layer. If we presume that the distillation process is symmetric, we expect the new model to perform better than the student model, and nearly as good as the original one we started with. We trained 10 models going from 1 to 10. The results can be seen in **Figure 2.**

We can see the accuracy is significantly better than our best student model with an average accuracy of around 0.84. However, it is not even close to the original accuracy of 0.9473.

It is also important to note that there is no significant deviation among the performance of the models, meaning the temperature parameter has little to

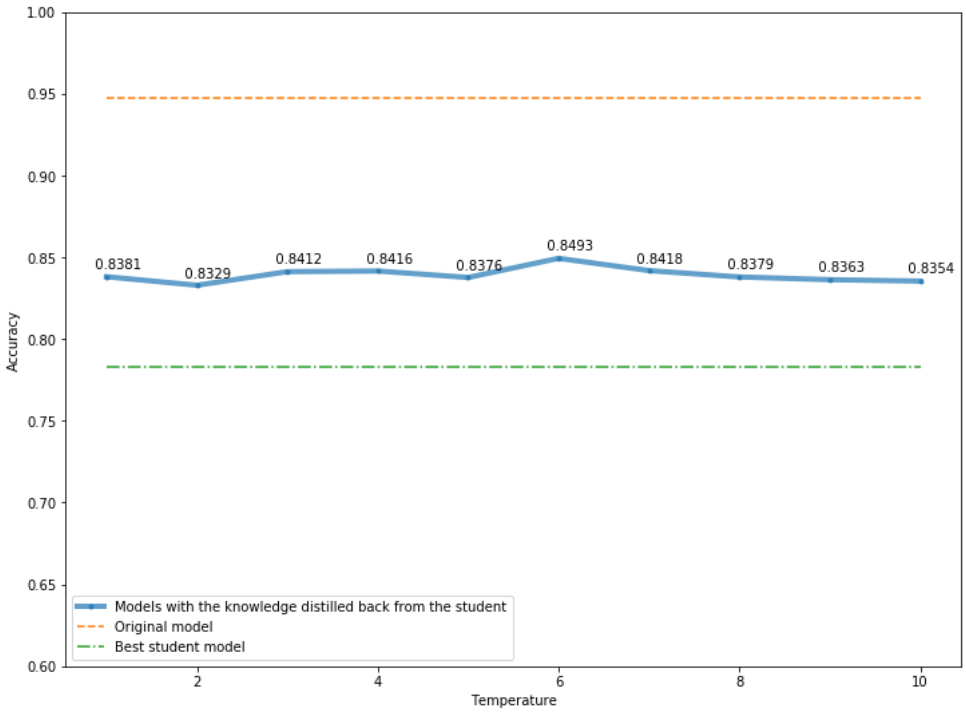

FIGURE 2. Results of distilling the knowledge back to the original model

no influence on the accuracy of the given model, and using this special parameterized softmax function provides no improvement as opposed to training traditionally on the output of the student model. With all that said, we can conclude that this method does not in fact keep symmetry.

4.2. **Student ensembles.** In this paragraph we are discussing whether we can improve the accuracy of the network in which we distilled the knowledge to, by creating an ensemble of networks of the same architecture, but using different distilling temperature parameter. To generate the predictions of the ensembles, we used a simple majority voting.

Using all 29 student networks, after evaluation, we achieved an overall accuracy of 0.7712 on our test set, which is certainly worse than our best student network (0.7827), but better than the average accuracy (0.65014). In order to improve this, we evaluated the best N networks. The results are shown in **Figure 3**.

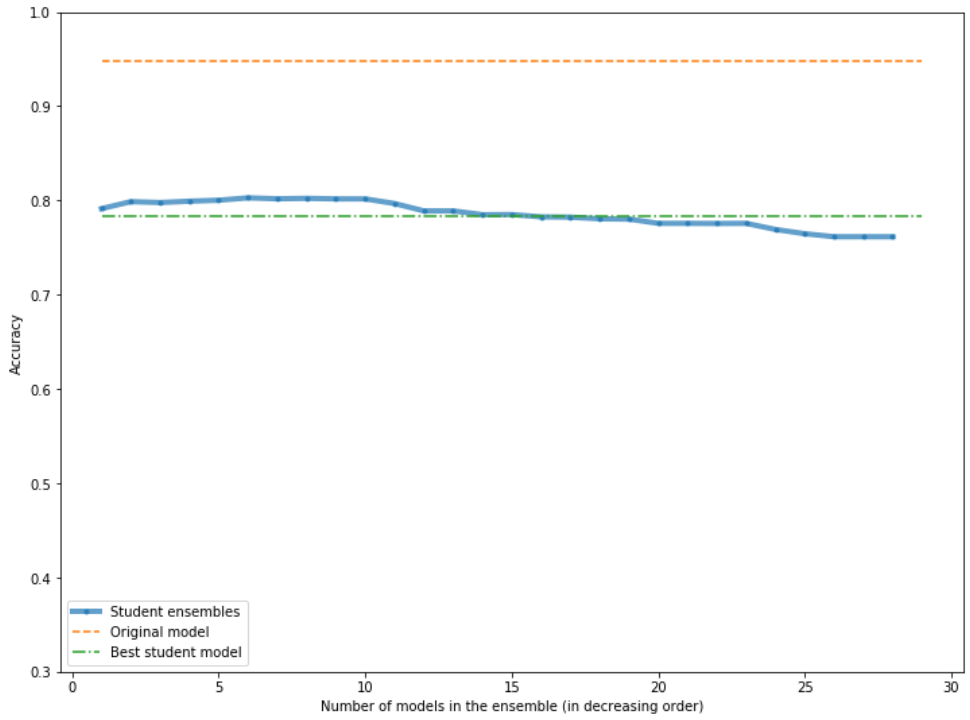

FIGURE 3. Results of distilling the knowledge back to the original model

Here we can clearly see that after 10 networks the accuracy is slowly, but steadily declining, and after 18 it goes below the best result of our individual student networks. It might be due to the fact that the models in our ensemble are structurally quite similar (the only difference is the temperature parameter) and the fact that they all were trained on the same data results in models that mostly make the same mistake during classification.

Considering that even if we find the best student models in relation to the temperature, then find the ideal number of networks for the ensemble and increase the complexity of the model, the boost in accuracy is not significant enough for this kind of trade off.

4.3. **Transitivity.** To test the transitivity of this technique, we first created a new model structure, which stood between the teacher and the student model in terms of complexity. It is a deep neural network with 2 hidden layers, with 50 nodes each. Then we distilled the knowledge with the discussed technique to the middle model.

21 different models were trained; one in the traditional way with hard outputs, and 20 with distillation with the temperature parameter ranging from 1 to 20. We then took the best performing model and distilled its knowledge further to the original student model.

Ideally, these results are comparable to the ones we received from directly distilling the knowledge to the student model. The results of the performance of the middle models in relation to the temperature can be seen in **Figure 4** (0 being the one trained on hard outputs).

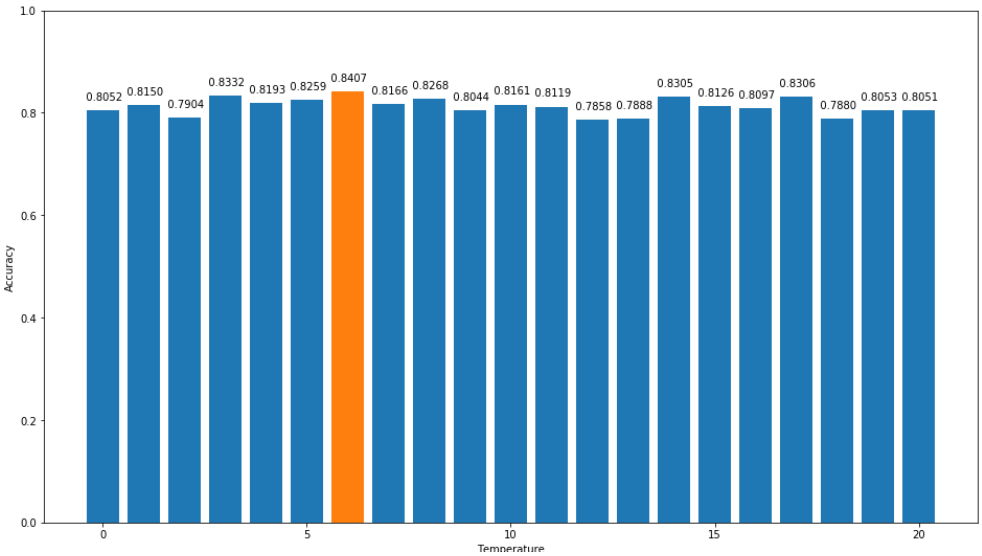

FIGURE 4. Results of distilling the knowledge to the middle model

After the experiment we saw that the best performing model was the one with temperature 6, which we used further in this experiment. Interesting to note that the improvement provided by the distillation method was insignificant as the performance of the non-distilled model was just slightly lower than the average of the distilled ones with very little standard deviation.

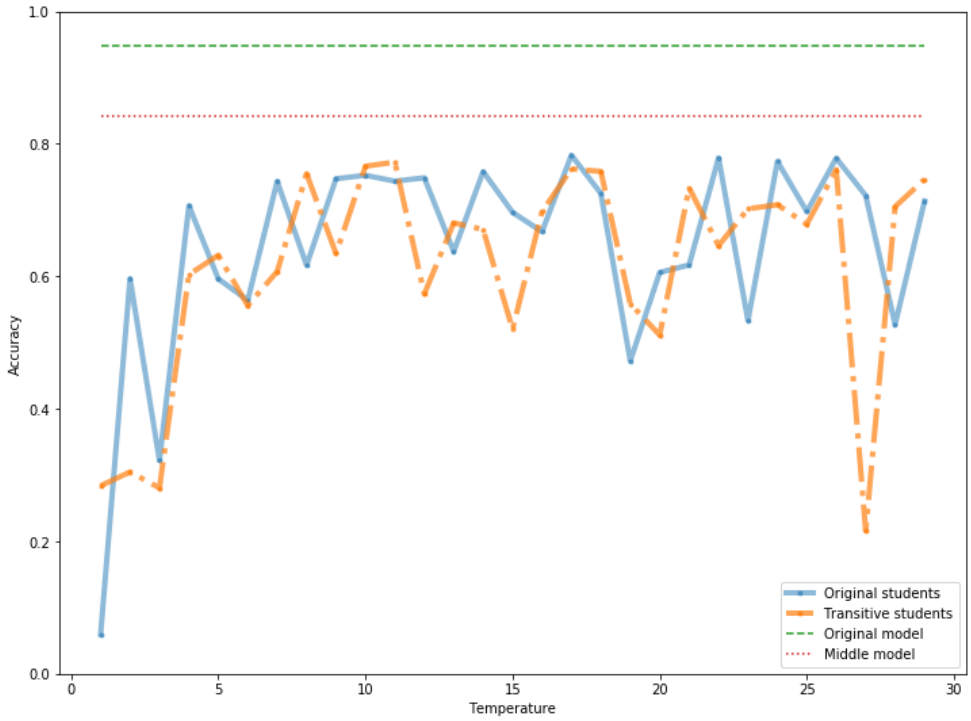

FIGURE 5. Results of the distillation

In **Figure 5** we can see the accuracy of the transitively distilled models in relation to the originals, as well as the baseline model and our highest performing middle model. Although the numbers are not exact, the overall distribution of the results are actually quite similar to the ones we had from the direct distillation.

We can claim that as long as the middle model achieves good enough accuracy – close to the original one – this distillation method keeps the transitive property.

5. Conclusion

In this paper we investigated multiple features and behaviours of the knowledge distillation method. We experimented with symmetry by distilling the knowledge back from our trained student network to our untrained teacher network. We conclude that even though it outperformed the best student, it did not come close to the model trained in the traditional way, and acted more as a noise rather than useful additional information, proving that this method is not symmetric.

Experiments for creating an ensemble of student networks were also conducted by using student networks trained with different temperatures. We were able to achieve very little improvement, which is due to the fact that besides the temperature there were no structural differences between the models, which resulted in similar cases of misclassification in every net. This leads us to believe that even though the temperature can affect the performance of the model, it has little to no effect on the behaviour.

Lastly, we investigated the transitive feature of this method by distilling the knowledge to a slightly more complex model than our student model, then distilled it further to our original student model. According to our experiment, the difference was not remarkable between these students and our baseline students, proving that this method is transitive.

6. Acknowledgements

The project has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002).

References

[1] Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001, 1990.

[2] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

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

[4] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4133–4141, 2017.

[5] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016.

[6] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

[7] Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

[8] Mengya Gao, Yujun Shen, Quanquan Li, and Chen Change Loy. Residual knowledge distillation. *arXiv preprint arXiv:2002.09168*, 2020.

[9] Mohammad Farhadi Bajestani and Yezhou Yang. Tkd: Temporal knowledge distillation for active perception. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 953–962, 2020.

[10] Xu Cheng, Zhefan Rao, Yilan Chen, and Quanshi Zhang. Explaining knowledge distillation by quantifying the knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12925–12935, 2020.

[11] Xiang Wu, Ran He, Yibo Hu, and Zhenan Sun. Learning an evolutionary embedding via massive knowledge distillation. *International Journal of Computer Vision*, pages 1–18, 2020.

[12] Inseop Chung, SeongUk Park, Jangho Kim, and Nojun Kwak. Feature-map-level online adversarial knowledge distillation. *arXiv preprint arXiv:2002.01775*, 2020.

[13] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[14] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[15] Kouichi Yamaguchi, Kenji Sakamoto, Toshio Akabane, and Yoshiji Fujimoto. A neural network for speaker-independent isolated word recognition. In *First International Conference on Spoken Language Processing*, pages 1077–1080, 1990.

[16] Maximilian Riesenhuber and Tomaso Poggio. Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019–1025, 1999.

[17] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[18] Jimmy Ba and Brendan Frey. Adaptive dropout for training deep neural networks. In *Advances in neural information processing systems*, pages 3084–3092, 2013.

[19] John S Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer, 1990.

[20] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32:323–332, 2012.

Department of Information Systems, Faculty of Informatics, ELTE Eötvös Loránd University, Budapest, Hungary

Email address: c4442f@inf.elte.hu, lkp@caesar.elte.hu, kiss@inf.elte.hu