Les Cahiers du GERAD ISSN: 0711–2440

Uncertainty transfer with knowledge distillation

I. Amara,
J. J. Clark

G–2020–23–EIW10

April 2020

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2020
– Bibliothèque et Archives Canada, 2020

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2020
– Library and Archives Canada, 2020

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Uncertainty transfer with knowledge distillation

Ibtihel Amara
James J. Clark

Centre for Intelligent Machines, McGill University,
Montréal (Québec), Canada, H3A 2A7

iamara@cim.mcgill.ca
clark@cim.mcgill.ca

April 2020
Les Cahiers du GERAD
G–2020–23–EIW10

Abstract: *Knowledge distillation is a technique that consists in training a student network, usually of a low capacity, to mimic the representation space and the performance of a pre-trained teacher network, often cumbersome, large and very high capacity. Starting from the observation that a student can learn about the teacher's ability in providing predictions, we examine the idea of uncertainty transfer from teacher to student network. We show that through distillation, the distilled network does not only mimic the teacher's performance but somehow captures the original network's uncertainty behavior. We provide experiments validating our hypothesis on the MNIST dataset.*

1 Introduction

The complexity of DNNs demands model compression. It is certain that training these very deep networks can be done with high performance CPUs or even with multiple GPUs. However, these trained models face a challenging situation during deployment wherein they are supposed to fit in terms of memory in small sensors, portable devices and are assumed to provide real-time predictions especially for critical applications such as decision making for autonomous driving. *Model compression* is a method of obtaining smaller and lower memory networks, trained to mimic the behaviour of the original trained large networks. There are different ways to achieve model compression in the literature. Some approaches concentrated on network pruning such as Weight quantization [9] and Low-rank approximation [19]. Other methods focused on network distillation [3, 10, 13, 16]. The latter is a method involving a teacher-student training. The student network, usually a shallow one with few parameters, is trained to replicate the performance of the teacher network, a very deep network. Over the past years, there have been many uncertainty estimation methods for Deep Learning models proposed. Since these deep models are being used for complicated tasks and risk-related applications, estimating uncertainty in these models has become as important as achieving high performance and accuracy. There are different types of uncertainty in Deep Learning [11] and there are various ways of extracting these from models [6, 18]. What all these trends have in common is that they focus on extracting uncertainty using the original network, which can often be very slow. However, in applications where execution speed and memory are crucial such in mobile devices, there should be an alternate way to acquire uncertainty without going through the original large network. In this work, we investigate whether distillation training results into uncertainty transfer from the teacher to the student network. The idea is that a student trained through the distillation process could be able to estimate the teacher network's capacity in providing prediction. In other words, the student can measure the teacher's uncertainty level given an input sample.

2 Related work

2.1 Knowledge Distillation (KD)

The distillation process involves teacher-student training. The student network usually consists of a smaller and shallower network (i.e. fewer parameters). It is trained to replicate the performance of the teacher network, which itself consists of a large and deep network. There have been different variants where different distillation losses were proposed. The work in [2] shows that it is possible to compress the information from an ensemble of networks into a single one. Some research looked at the architecture of the fully connected deep networks and distilled their behavior onto shallow but wider hidden units [1]. Hinton et al. [10] proposed a network that involves the use of parameter called the *temperature factor* at the level of logits. This temperature parameter determines how much the activations of incorrect classes are encouraged. There is also the work of Chen et al. [4] that involved the learning of a "function-preserving" transformation to initialize the parameters of a larger student network. Their goal was to achieve a faster training of deep neural networks.

2.2 Uncertainties in deep learning

There are different types of uncertainty in Deep Learning [11]: aleatoric, which captures the noise related to the observations, and epistemic, which captures uncertainty about the model parameters. Aleatoric uncertainty is the ambiguity about the observation caused by a noisy dataset. An example of aleatoric uncertainty in images would be due to occlusions. Epistemic uncertainty is associated with our incapacity to explain which model parameters are responsible for the set of model outputs. This uncertainty can be explained away if enough data is given to the model [6]. Epistemic uncertainty is important for critical applications and for models that are trained on very small datasets. There have been many methods to quantify uncertainty in Deep Learning. We can categorize these uncertainty measures into three main groups: True Bayesian framework, Bayesian Approximation framework, and the non-Bayesian framework. The Bayesian formalism naturally incorporates uncertainty modelling with probability distributions, which shows the degree of belief regarding of the unknown parameters. Exact Bayesian methods are known to be computationally expensive and complicated especially when dealing with millions of parameters [7]. Therefore, a variety of methods have been developed to overcome this problem, such as variational approximations. The most used techniques under the Bayesian approximation framework are the Monte-Carlo dropout [6] and Monte-Carlo batch normalization[18]. For the non-Bayesian framework for quantifying uncertainty, methods based on ensembles such as Deep ensembles [12] can be used.

3 Methods

3.1 Distillation

In this paper we use the method of knowledge distillation proposed by [10]. To train the distilled network, we are in need of both of hard targets, which are the true labels of the dataset, and the soft targets, which are the class probability produced by the teacher network. This method shows that the logits, which are the inputs to the final softmax, can be used for training small models. These soft targets are obtained through the softmax function that uses a temperature value. A higher value of this temperature parameter produces softer probabilities over classes. The distillation process is then performed through the minimization of the average of two objective functions: (1) cross entropy with the soft targets with high temperature for the softmax and (2) cross entropy with the correct labels using the logits in the softmax.

3.2 Model uncertainty with Monte-Carlo dropout

We used the method of MC dropout to capture model uncertainty. This method, in practice, consists of having a deep network trained using dropout. At test time, we use dropout to sample N networks and record their predictions. The variance of these outputs is the model uncertainty. Please refer to the original work [6] for more details on the mathematical formalism.

4 Experiments and results

4.1 Experimental set up

To investigate our hypothesis that distilled networks can estimate the model uncertainty of the original deep network, we conducted experiments where we considered a VGG16-like network, trained for classification with dropout (having a fixed probability of $p = 0.5$, since this is considered to be the optimal range [17]) using the MNIST dataset. We performed distillation training onto three different student networks in which we varied the number of learnable parameters. The teacher network has 16 layers with $14,913,226$ learnable parameters. The smallest student network (SN1) has four layers with $4,265,066$ parameters. The student network (SN2) has seven layers with $4,721,610$ parameters. Finally, student network 3 (SN3) has the exact same architecture as the teacher network (i.e. 16 layers

and 14, 913, 226 parameters). To evaluate the captured model uncertainty of these different student networks we use these following measures.

Calibration Curves To see if a model is well calibrated, we often use *calibration curves*, sometimes called *reliability diagrams* [5, 15]. These diagrams are a way to represent model calibration. They represent expected accuracy as a function of the confidence (i.e. softmax probabilities). A "perfectly" calibrated model is represented by the identity function. A deviation above (under-confidence) or below (over-confident) the diagonal shows a miscalibrated model. To plot this curve, the confidence level (i.e. probability predictions of the network) are regrouped into M bins of size 10.

Expected Calibration Error In contrast to the calibration curves, which are visual diagrams capturing the degree of model calibration, the Expected Calibration Error (ECE) [14] is a loss function that returns a scalar value that can capture the degree of miscalibration of a particular deep learning model. Similarly to the calibration curves, confidence values are partitioned into $M = 10$ equal bins and we take the weighted average of the difference between the accuracy and confidence at each bin. Full details about this metric can be found at [8].

Mean Uncertainty Prediction Error This measures the mean squared error between the predicted confidence (uncertainty) of the teacher and student networks over the test dataset.

4.2 Results and discussion

We provide in Table 1 the classification accuracy for each of the teacher and student networks trained with KD (KD-SN1, KD-SN2, and KD-SN3) and without KD for student network 2 (SN2). We can observe that each of the distilled student network was able to closely represent and mimic the teacher's performance on all sets of the MNIST dataset. In fact, as these student models get deeper and larger, we can note a decrease in the accuracy. This decrease within the different student networks could be explained by the fact that the MNIST dataset does not require very deep models to achieve high performance. Instead, a smaller network can perform better. Through the use of soft targets, we would expect the larger student network (KD-SN3) to have at least the closest performance to the teacher network since they share the same architecture and capacity. However, we see that distilling a large network onto a large student network does not always lead to similar performance. The student networks SN2 trained without KD and KD-SN2 trained with KD have similar performance on the dataset.

Table 1: Summary table of Classification Accuracies for all networks, Expected Calibration Error (ECE), and Mean Uncertainty Prediction Error (MUPE)

	TN	KD-SN1	KD-SN2	SN2	KD-SN3
Training acc.	99.87%	99.75%	99.52%	99.99%	98.75%
Validation acc.	99.08%	98.93%	98.80%	99.44%	98.08%
Test acc.	99.21%	99.14%	99.04%	99.51%	98.36%
ECE	0.0027	0.0064	0.0019	0.0036	0.0029
MUPE	—	0.0037	0.0019	–	0.0016

One question that can be asked here is: *If we could train a student network to perform as accurate as the teacher model (in giving similar class accuracy) can this student exhibit similar uncertainty behavior as the teacher network?*

To attempt to provide an answer to this question, we performed 100 Monte Carlo (MC) dropout at test time on each of the teacher and student networks then record their predictive mean confidences and plot their corresponding calibration curves. Figures 1, 2, and 3 show, respectively, the calibration graphs MC KD-SN1, MC KD-SN2, and MC KD-SN3 overlayed on the MC TN calibration curve. We chose to plot the MC networks to better apprehend the model uncertainty behavior that underlies within each of these networks.

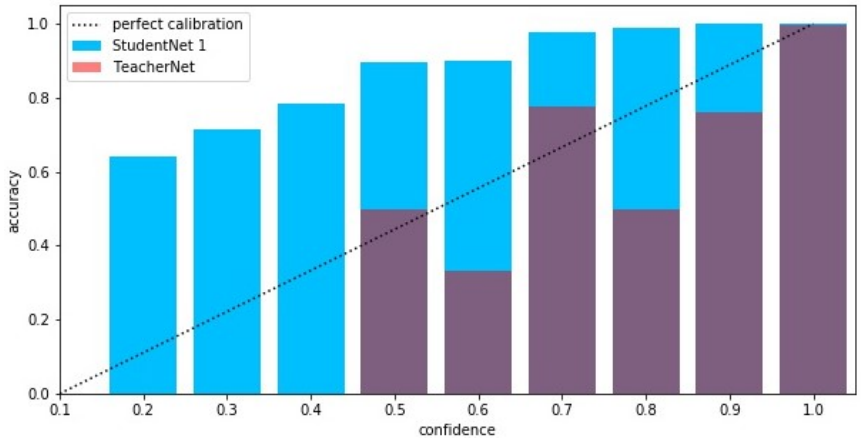

Figure 1: Calibration curve of the MC Student Network 1 (light blue) overlaid on the calibration of MC Teacher Network (red)

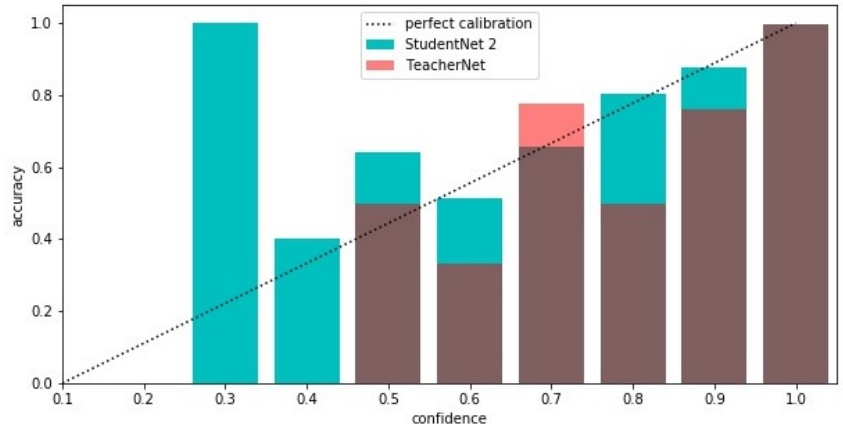

Figure 2: Calibration curve of the MC Student Network 2 (blue) overlaid on the calibration of MC Teacher Network (red)

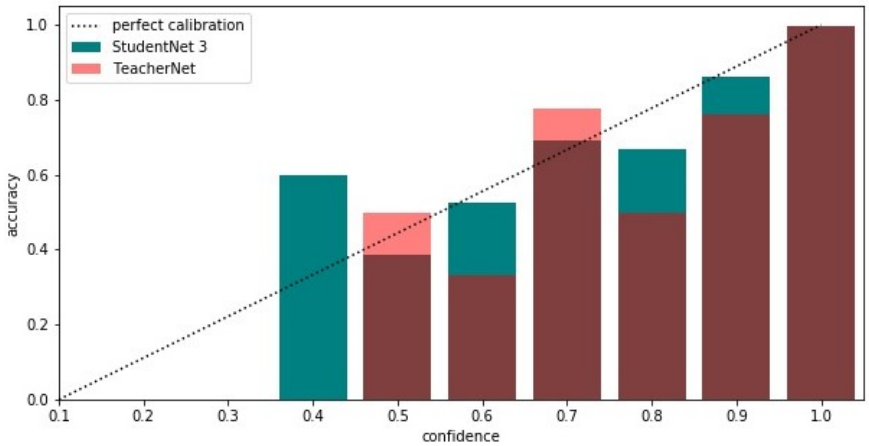

Figure 3: Calibration curve of the MC Student Network 3 (teal) overlaid on the MC Teacher Network (red)

By comparing these graphs, we could observe that as the student network gets deeper and asymptotically closer to the number of parameters of the original network, the calibration curves tend to be as similar as the teacher network. We observe from the figures that the calibration curve of student SN3 is visually closer to the teacher network's calibration graph. This observation can also be analyzed

using the statistical metric ECE. In Table 1, we provide the ECE for each of the teacher and student networks. We see that both student networks KD-SN2 and KD-SN3 are closer to the miscalibration value of the teacher. Specifically, the ECE of KD-SN3 is the closest to the ECE of the teacher network. It is also noteworthy to mention that the ECE of KD-SN2 can also be considered as a close value to the ECE of the teacher model even though this has got better model calibration. Furthermore, the mean uncertainty prediction error can be observed in Table 1 for each of the MC student networks. We note that student KD-SN3 has the lowest mean uncertainty error, which shows that this network reflects the best on the uncertainty behavior of the teacher network. Although both student networks KD-SN2 and KD-SN3 did not provide the closest class accuracy, these distilled networks somehow reflect the teacher's uncertainty. At this stage, we could say that there is a partial uncertainty transfer from teacher to student through the distillation process. However, there are some factors associated with this uncertainty transfer which are mainly the distilled network architecture configuration and the network capacity. Moreover, we see in Table 1 that the ECE for KD-SN2 is lower than the ECE of SN2. This shows that the student network trained with KD is better calibrated. As we have seen, a large student network can almost capture the teacher's uncertainty. Nevertheless, a smaller and compressed network can be used as a higher level insight about the teacher model's uncertainty behavior. Additionally, KD training can be useful for model calibration.

5 Conclusion

We investigated in this paper the possible uncertainty transfer between teacher and student network during distillation training. Our key finding is that uncertainty transfer through distillation can not happen when the student network's capacity is too low.

References

[1] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In Advances in neural information processing systems, pages 2654–2662, 2014.

[2] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 535–541. ACM, 2006.

[3] Zhengping Che, Sanjay Purushotham, Robinder Khemani, and Yan Liu. Distilling knowledge from deep networks with applications to healthcare domain. arXiv preprint arXiv:1512.03542, 2015.

[4] Wenlin Chen, James Wilson, Stephen Tyree, Kilian Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In International Conference on Machine Learning, pages 2285–2294, 2015.

[5] Morris H DeGroot and Stephen E Fienberg. The comparison and evaluation of forecasters. Journal of the Royal Statistical Society: Series D (The Statistician), 32(1–2):12–22, 1983.

[6] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In international conference on machine learning, pages 1050–1059, 2016.

[7] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pages 1183–1192. JMLR. org, 2017.

[8] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pages 1321–1330. JMLR. org, 2017.

[9] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149, 2015.

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

[11] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In Advances in neural information processing systems, pages 5574–5584, 2017.

[12] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In Advances in Neural Information Processing Systems, pages 6402–6413, 2017.

[13] David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information. arXiv preprint arXiv:1511.03643, 2015.

[14] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015.

[15] Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In Proceedings of the 22nd international conference on Machine learning, pages 625–632. ACM, 2005.

[16] George Papamakarios and Iain Murray. Distilling intractable generative models. In Probabilistic Integration Workshop at Neural Information Processing Systems, 2015.

[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] Mattias Teye, Hossein Azizpour, and Kevin Smith. Bayesian uncertainty estimation for batch normalized deep networks. arXiv preprint arXiv:1802.06455, 2018.

[19] Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 7370–7379, 2017.