What is Lost in Knowledge Distillation?

Manas Mohanty **Tanya Roosta** **Peyman Passban**
mhmanas@amazon.com

Abstract

Deep neural networks (DNNs) have improved NLP tasks significantly, but training and maintaining such networks could be costly. Model compression techniques, such as, knowledge distillation (KD), have been proposed to address the issue; however, the compression process could be lossy. Motivated by this, our work investigates how a distilled student model differs from its teacher, if the distillation process causes any information losses, and if the loss follows a specific pattern. Our experiments aim to shed light on the type of tasks might be less or more sensitive to KD by reporting data points on the contribution of different factors, such as the number of layers or attention heads. Results such as ours could be utilized when determining effective and efficient configurations to achieve optimal information transfers between larger (teacher) and smaller (student) models.

1 Introduction

DNNs have contributed significantly to enabling advances in many tasks, including NLP. In this context, *deep* refers to a large number of neural parameters. With the advancements in the computing hardware, training of DNNs with billions of parameters is now feasible. However, it does not necessarily mean that utilizing them for different setups has become easier. The computational complexity, storage requirements, and added latency still make their deployment in real-world difficult. This issue is more pronounced on the ever-smaller devices, with limited processing and storage capacity.

To make DNNs useful for low-budget settings, there are a number of approaches have been developed for model compression Cheng et al. (2017), which fall under one of the following categories of *Pruning* (Blalock et al., 2020), *Quantization* Courbariaux et al. (2015); Sindhwani et al. (2015); Wu et al. (2015); Zhai et al. (2016), or knowledge distillation (KD) (Sanh et al., 2019; Gou et al., 2020; Passban et al., 2020). Our work focuses on KD in the hope of understanding what part of knowledge is transferred during distillation. Specifically, the research question we aim to answer is *what type of information is lost during the distillation process?* It might not be possible to fully dissect this concept but it is defiantly worthwhile to understand what classes, sets of tasks, and parameters are impacted more than others during distillation, so that the right mitigation can be put in place to account for the loss.

2 Background

KD refers to transferring knowledge from a bigger or deeper model, called teacher ($\mathcal{T}$), to a smaller or shallower model, called student ($\mathcal{S}$). The term was first coined in Bucila et al. (2006), and then elaborated by Hinton et al. (2015), where KD was presented as a compression framework. Compression is achieved by a student model imitating its teacher's output. The output of a DNN is usually a class probability:

$$p(z_i, T) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$$

where z_i is the *i*-th class's logit and T is called the temperature. The loss function for distillation is usually formulated as:

$$\mathcal{L}_{KD} = Loss(p(z_t, \mathcal{T}), p(z_s, \mathcal{S}))$$

where the *Loss* function is often a KL divergence Gou et al. (2020), and z_s and z_t are the logits for the student and teacher models, respectively. The total loss is then an interpolation of the task loss ($\mathcal{L}_{task}$), and the KD loss:

$$L = \alpha \times \mathcal{L}_{task} + \eta \times \mathcal{L}_{KD}$$

with α and η (usually $\eta = 1 - \alpha$) as the weight values to indicate the contribution of each term to the learning process.

3 Related Work

According to Gou et al. (2020), KD approaches can be categorized into the following types of: *i*) Response-based KD, where the logits of the output layers are matched between the teacher and the student models (Sanh et al., 2019); *ii*) Feature-based KD, where the outputs of specific intermediate hidden layers as well as those of the final layers are used to train the student model (Sun et al., 2019; Wu et al., 2020); and *iii*) Relation-based KD which is similar to the feature-based approach as both the intermediate and final output layers are utilized in training, but instead of using specific layers, a relationship between different layers is explored (Passban et al., 2020). In this paper, we use the response-based and feature-based KD approaches in our experiments, which enable us to keep the setup more manageable to investigate the problem. Relation-based KD introduces multiple moving/tunable factors which makes it enigmatic to justify and understand if the losses captured in the experiments are rooted in KD or in layer-to-layer communications.

The research on the topic of KD is of great interest to many, and there are myriad of prior work on various aspects of it. However, to the best of our knowledge, the following papers are the closest to our research point of view. Hooker et al. (2019) looked at the question of what compressed neural networks forget. They ask the question of whether the test accuracy is the best measure to determine whether the compressed model can generalize. They looked at the cases where the top level performance metrics are similar between the base and compressed model; however, some of the classes are dis-proportionally impacted by the compression. They tried to understand what makes some of the classes more sensitive to performance degradation. Their work specifically focused on image recognition and is on the pruning and quantization sides, not KD, but the way they probed DNNs is conceptually similar to our investigating.

Michel et al. (2019a) looked at the attention-based models in NLP and tried to determine if all attention heads are necessary when making predictions. They made an observation that for a set of tasks, during inference time, having sixteen heads for attention is not necessarily better than having only one attention head. This form of investigation is also similar to what we carried out in our research.

In Wang et al. (2020), the authors proposed a KD method for Transformers which is task-agnostic. Their method does not require a certain architecture guarantee for the student model and showed improvements on various GLUE tasks. The task-independent nature of this work makes it suitable to investigate the behaviour of KD methods.

To the best of our knowledge, there has been no work performed on determining how KD affects different inference tasks, and how it affects classes with different labels. In this paper, we focus on these research questions. We believe, this could be of great use since existing NLP models are typically large, but the real-world use cases dictate limited resources. Understanding what information is lost (or retained) for a given configuration in a given task, can help guide system designers optimize their setup.

4 Models and Experimental Hypotheses

For our experiments, we use a 12-layer RoBERTa base model (Liu et al., 2019) as the teacher. It is a well-studied and widely used model in the NLP community, which makes the reproduction of our work easier for other researchers. In each distillation setup, we keep all the configuration parameters untouched apart from the one that is of interest, e.g. we fix all the configuration parameters except

the number of hidden layers. Then, we distill the base model into different student models each with a different number of hidden layers. The same approach is used for all the other configuration parameters. This leads to training of 3L, 6L, 9L, 4AH, 8AH, 384D, 516D, and 6L_384D students, where L, AH, and D stand for Layer, Attention Head, and Dimension, respectively, i.e. 9L refers to a model that is identical to its teacher but it only has 9 layers.

When reducing the size of the teacher model (from 12 to fewer layers), we had to come up with a mapping strategy to carry out the layer-to-layer distillation. After investigating a comprehensive set of mappings, the solution we arrived at was to always connect the first and last layers of the student and teacher models in all configurations. For internal layers, in 6L, with the exception of the third layer we skipped every other teacher layer and connected the rest uniformly. For 3L, we connected the fifth layer of the teacher to the middle layer of the student. In 9L, we connected the first 2 and the last 5 layers of the student to the first 2 and last 5 layers of the teacher, and aligned the third and fourth student layers to the fourth and sixth teacher layers, respectively. Similar to the number of layers, we also investigated the impact of tighter layers by shrinking the widths of the internal units to 516 and 384. We evaluated the distilled models over 8 tasks from GLUE (Wang et al., 2018), using the standard training and development sets from the various GLUE tasks.

In regard to hyper-parameter, for distilling *all configurations*, the hard, intermediate, and KD loss weights are each equally set to 0.33. The temperature is set to 2 and each batch processes 12 instances. For fine-tuning, the learning rate of $5e-5$ worked best. Adam (Kingma and Ba, 2014) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ was used as the optimizer, and the number of epoch for fine-tuning was 10.

5 Experimental Results

Table 1 summarizes all our observations. These numbers can be interpreted from different perspectives and can vary in different settings. However, our findings show that: SST2 and QQP are the most resistant tasks (datasets) in the presence of KD whereas RTE is the most sensitive one. CoLA is also impacted severely since it loses most of the information through the KD process, across all configurations. QNLI, MRPC, and STSB are only impacted slightly and behave similarly. MNLI is in the middle of these two extremes; KD impacts it to some extent but not as much as the sensitive group. Apart from MNLI, there is a direct relation between the size of the training set and the KD loss, i.e. the smaller the set, the higher the loss. However, MNLI breaks this pattern as it has the largest dataset among all but still shows relatively high losses. This could be due to the complex nature of the inference task. It is the only dataset with 3 classes, compared to others with ≤ 2 classes.

	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB
baseline	0.62	0.88	0.92	0.93	0.9	0.8	0.95	0.91
9L	0.59	0.83	0.89	0.9	0.89	0.68	0.93	0.87
6L	0.6	0.83	0.87	0.9	0.88	0.69	0.91	0.88
3L	0.4	0.77	0.81	0.85	0.83	0.58	0.91	0.81
516D	0.55	0.81	0.87	0.88	0.89	0.62	0.9	0.87
384D	0.53	0.8	0.88	0.87	0.87	0.65	0.9	0.86
8AH	0.55	0.81	0.87	0.88	0.88	0.61	0.92	0.85
4AH	0.55	0.81	0.88	0.88	0.89	0.61	0.92	0.85
6L_384D	0.47	0.78	0.85	0.84	0.87	0.62	0.91	0.84

Table 1: The performance scores of KD models. STSB and CoLA use Pearson-Spearman and Matthews correlation coefficients. MRPC and QQP use the average of Accuracy and F1 scores and others rely on accuracy as their evaluation metric.

Reducing the number of layers up to a certain threshold is the best form of KD with only marginal losses. Both 9L and 6L show acceptable losses while reducing the number of layers to 3 was a drastic change and impacted the model significantly. The loss introduced by modifying the width of the hidden layers depends on the nature of the task and the dataset, but our results indicate that it can still be considered as a reasonable KD alternative. The loss is defiantly larger than that of the layer

reduction but still is in an acceptable range. Combining both ideas, namely reducing the number and width of layers together is harmful and causes serious deterioration.

Different researchers (Michel et al., 2019b; Parnami et al., 2021) have previously reported that the number of attention heads can be reduced with almost *no significant impact* on final results. With this assumption, we were expecting marginal losses after changing the number of heads but it did not happen in our case. We noticed that the number of heads plays a critical role in KD.

By investigating more data points and different aspects of the results, we hope to find a pattern for losses introduced by KD. We thus produced a heatmap of student-teacher disagreements in Figure 1 based on the results from Table 1. As the figure shows, the highest losses belong to *3L* models and more challenging datasets are *RTE* and *CoLA* on average. The heatmap simply visualized which tasks or models behave similarly.

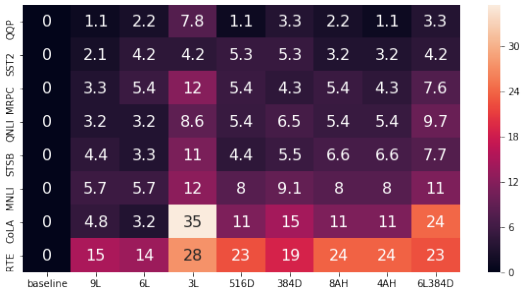

Figure 1: How much distillation degrades the student model. The intensity of the color of the cells (and numbers inside) show the percentage of disagreement between the student and teacher models, e.g. [CoLA][3L] = 35 indicates that 3L is 35% weaker than its teacher on the CoLA dataset.

5.1 Response Time

KD is a compression strategy so we expect faster engines as the result of this process. To study how KD impacts models' response time we measured the number of samples per second each model can process under a common setting. Results of this experiment are reported in Table 2. We used a p3.8x environment with a set of four V100 GPUs (from AWS). Across all configurations and models we see a faster response time. However, students with fewer attention heads break this pattern. In theory, reducing the number of heads should also help with the speed increase whereas this is not observed in our experiments. Multiple factors, such as implementation techniques, hardware, non-optimal matrix multiplication etc can lead to this result, for example, in Transformers, parallelism mainly comes from the multi-head attention mechanism, where different heads can attend to different parts of the input sequence simultaneously. Each attention head processes a different subspace of the input, which allows for parallelization. Therefore, reducing the number of attention heads can reduce the scope of parallelism, which might lead to slow down in inference.

	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB
baseline	49.1	185.8	172	184.5	186.7	165.6	180.6	181.3
9L	50.4	184.2	175.8	183.7	185.3	170.6	182.5	183.8
6L	56	278.1	262.3	277.5	280.4	260.5	260.1	275.5
3L	57.3	324.2	305	321.2	324.5	301.9	318.4	316.4
516D	51.9	210.4	195.6	210.1	211.1	191	205.6	208.6
384D	54	262	240.8	260.3	265.2	231.5	253.5	256.9
8AH	46.9	152.7	139.3	153.6	154.2	140.5	150.8	153
4AH	47.8	153.9	138.5	152.3	154.2	133.6	149.2	150.3
6L_384D	59	408.4	387.1	403.7	408.7	379.2	389	397.7

Table 2: The number of samples processed by each model per second.

5.2 Aggregating Quantitative Results

We aggregated results from Tables 1 and 2 to be able to design a distillation recipe based on quality and speed gains of student models, e.g the loss for both 3L and 6L_384D is comparable but considering the faster response time, 6L_384D seems to be a better alternative. We incorporated our findings and summarized them in Table 3, which aims to facilitate selecting the right KD configuration in different applications. The ✓, ✗, and ? signs in the table indicate whether KD is worthwhile or not. For example, [Layer][CoLA] = ✓ means no matter which layer-reduction strategy we use, the loss we get from KD is almost consistent, thus we can lean towards a higher compression rate by eliminating more layers and distillation is worthwhile. On the contrary, ✗ shows that the KD approach is harmful. Unlike the two aforementioned signs, ? is a sign of uncertainly, meaning KD may or may not succeed and it depends on the setup.

Comparing these factors side-by-side could justify whether KD is a reasonable option or not, i.e from Table 3, STSB is immune to all configuration modifications but the attention heads. That means, we can safely reduce the number of layers or shrink the hidden layers yet expect similar high-quality results. From a response-time perspective (shown with *Speed* in the table), it is also worth considering KD for STSB. However, modifying the number of attention heads can yield some inconsistencies, so we would consider KD for STSB only when it is possible to keep the number of attend heads untouched. The table only shows results for CoLA, RTE, and STSB. For the other five tasks, the signs are ✓ across all factors.

	CoLA	RTE	STSB
Layer	✓	?	✓
Att. Head	?	✗	?
Hidden Dim	✗	✗	✓
Speed	?	✓	✓

Table 3: Scenarios in which KD could be lossy.

6 Conclusion and Future Work

In this paper, we designed a set of experiments to better understand what sort of information is likely to be lost during the distillation process. We focused on different factors, such as, the number of layers or the width of each layer, and showed how they affect the final performance. Our observations showed that in general, the number of layers is not that sensitive to KD, as long as there is no drastic reduction. The width of the hidden layers, on the other hand, showed the highest sensitivity. This feature seems to be quite fragile, and once changed, the final performance changes proportionally. The number of attention heads also shows high sensitivity. We could gain some insight about the behaviour of distilled models via our experiments, but we (as the community) are still far from decoding the problem and deriving conclusions. In our future work, we plan to run more experiments with different teachers, and expand our experiments' domain by testing more datasets and tasks. We are in the era of large language models, and smaller models do not receive enough attention. However, they still have their applications, and most of real-world problems can be solved effectively and efficiently by the right choice of such compact models.

References

Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. 2020. What is the state of neural network pruning? In *Proceedings of Machine Learning and Systems*, volume 2, pages 129–146.

Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Knowledge Discovery and Data Mining*.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. 2017. A survey of model compression and acceleration for deep neural networks. *CoRR*, abs/1710.09282.

Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. 2015. Binaryconnect: Training deep neural networks with binary weights during propagations. *CoRR*, abs/1511.00363.

Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. 2020. Knowledge distillation: A survey. *CoRR*, abs/2006.05525.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Sara Hooker, Aaron C. Courville, Yann N. Dauphin, and Andrea Frome. 2019. Selective brain damage: Measuring the disparate impact of model pruning. *CoRR*, abs/1911.05248.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Paul Michel, Omer Levy, and Graham Neubig. 2019a. Are sixteen heads really better than one? *CoRR*, abs/1905.10650.

Paul Michel, Omer Levy, and Graham Neubig. 2019b. Are sixteen heads really better than one? In *NeurIPS*.

Archit Parnami, Rahul Singh, and Tarun Joshi. 2021. Pruning attention heads of transformer models using a* search: A novel approach to compress big NLP architectures. *CoRR*, abs/2110.15225.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2020. ALP-KD: attention-based layer projection for knowledge distillation. *CoRR*, abs/2012.14022.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Vikas Sindhwani, Tara N. Sainath, and Sanjiv Kumar. 2015. Structured transforms for small-footprint deep learning.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers.

Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Jian Cheng. 2015. Quantized convolutional neural networks for mobile devices. *CoRR*, abs/1512.06473.

Yimeng Wu, Peyman Passban, Mehdi Rezagholizadeh, and Qun Liu. 2020. Why skip if you can combine: A simple knowledge distillation technique for intermediate layers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1016–1021, Online. Association for Computational Linguistics.

Shuangfei Zhai, Yu Cheng, Weining Lu, and Zhongfei Zhang. 2016. Doubly convolutional neural networks. *CoRR*, abs/1610.09716.

7 Appendix

Student Models: Table 4 provides detailed information about the student models and their configuration.

Model	Layer	Attention Heads	Layer Dimension.	Parameter
Teacher	12	12	768	124M
9L	9	12	768	103M
6L	6	12	768	82M
3L	3	12	768	61M
8AH	12	8	768	124M
4AH	12	4	768	124M
516D	12	12	516	65M
384D	12	12	384	41M
6L_384D	6	12	384	30M

Table 4: Distillation configurations used in our experiments. RoBERTa base is the teacher model. The last columns shows the number of total parameters of each model.

Datasets: Table 5 provides detail information about experimental datasets.

Corpus	Task	Train	Dev
MRPC	Paraphrase	3.7k	408
RTE	NLI	2.5k	276
QNLI	QA/NLI	108k	5.7k
CoLA	Acceptability	8.5K	1K
MNLI	NLI	393k	20k
SST2	Sentiment	67k	872
QQP	Paraphrase	364k	40k
STSB	Textual Similarity	5.7K	1.5K

Table 5: The statistics of the tasks/datasets from the GLUE collection (Wang et al., 2018). Apart from STSB which is a regression task and MNLI which has 3 classes, all the other tasks fall under binary classification.

Qualitative Analysis: In addition to the quantitative experiments reported in the paper, we ran a set of qualitative analyses. We mainly extracted four types of instances from our datasets where *i*) the teacher and all student models match the true label (TL); *ii*) the teacher and only a subset of performant students agree with TL; *iii*) only the teacher is able to produce TL, and finally *iv*) both the teacher and students fail to match TL. In total, we picked 140 instances and manually analyzed them, in the hope of finding common patterns in these instances. Table 6 shows a subset of our samples.

For the last group where both the teacher and all students fail to detect the correct class we could not find any explainable logic and it is not quite clear why even a strong teacher model cannot learn the task. However, for other categories we noticed some commonalities across different datasets. For trivial examples, all the teacher and student models easily solve the task because we believe *the input is compatible with the nature of the task*, i.e. in the CoLA example provided in the table, the task is to check the grammatical correctness of the input and the short length of it makes the job relatively straightforward. Similarly, in the second sample from QQP, the words "billion" and "Googol" carry strong signals which makes comparison of Q1 and Q2 easy.

For slightly complex examples where sentences are *long* or the wording varies in the input tuples, only student models with a higher learning capacity can succeed. The higher the capacity of the student, the higher its chance to mimic its teacher, e.g in the RTE example only the weakest model (3L) fails to predict the right class. Occurrence of common phrases such as "Two British", "clashes", or "police station" also makes compering S1 and S2 less complicated and students equipped with

Dataset	Example	TL	$\mathcal{T}$	9L	6L	516D	384D	3L
CoLA	Dana walked and Leslie ran	1	1	1	1	1	1	1
QQP	**Q1**: How many zeroes are there in one billion? **Q2**: How many zeroes does a Googol have?	0	0	0	0	0	0	0
RTE	**S1**: Two British soldiers have been arrested in the southern Iraq city of Basra, sparking clashes outside a police station where they are being held **S2**: Two British tanks, sent to the police station where the soldiers are being held, were set alight in clashes	1	1	1	1	1	1	0
QNLI	**Q**: What religion is the western region mostly? **A**: The upper part of Kenya's Eastern Region is home to 10% of the country's Muslims, where they constitute the majority religious group.	1	1	1	1	0	0	0
QQP	**Q1**: Which is the best laptop below rs60000? **Q2**: Which is the best laptop to buy under 50k?	0	0	1	1	1	1	1
MRPC	**S1**: While dioxin levels in the environment were up last year , they have dropped by 75 percent since the 1970s, said Caswell **S2**: The Institute said dioxin levels in the environment have fallen by as much as 76 percent since the 1970s	0	0	1	1	1	1	1
RTE	**S1**: Today's best estimate of giant panda numbers in the wild is about 1,100 individuals living in up to 32 separate populations mostly in China's Sichuan Province, but also in Shaanxi and Gansu provinces **S2**: There are 32 pandas in the wild in China	1	0	0	0	0	0	0

Table 6: A subset of samples studied in our qualitative analysis. TL stands for True Label and $\mathcal{T}$ is the teacher prediction.

better memory/learning modules can benefit from it. In the case of QNLI the situation is slightly more challenging. The way the same concept is phrased is different from Q1 to Q2 thus only (really) high-capacity models are able to judge correctly. Also, in this scenario, the very same short-length feature that was useful in CoLA hurts the QNLI model as a longer sequence (in Q) could provide the model with more context.

What we summarized here, clearly, does not universally apply to all KD models and datasets, and it is just our observation but it was interesting to see how there may exit a *common pattern* in the failed/successful cases. Moreover, we found out that regardless of the dataset, model architecture, and task, existing NLP models as well as KD techniques need serious (if not revolutionary) modifications. We encountered examples which are trivial to comprehend for us (as humans) but our best models failed to tackle. The last RTE example in the table could be one of them. Finally, we witnessed multiple cases of *memorization* rather than *learning*, the well-known shortcoming of neural models, where by changing a single word in a long sentence the behaviour of the model changes completely (as soon as the model is exposed to an unfamiliar input it fails to respond).