information

Article

Knowledge Distillation: A Method for Making Neural Machine Translation More Efficient

Wandri Jooste [1,*], Rejwanul Haque [2,*] and Andy Way [1,*]

[1] ML-Labs, ADAPT Centre, Dublin City University, D09 Y074 Dublin, Ireland
[2] School of Computing, National College of Ireland, D01 Y300 Dublin, Ireland
[*] Correspondence: wandri.jooste2@mail.dcu.ie (W.J.); rejwanul.haque@ncirl.ie (R.H.); andy.way@adaptcentre.ie (A.W.)

Abstract: Neural machine translation (NMT) systems have greatly improved the quality available from machine translation (MT) compared to statistical machine translation (SMT) systems. However, these state-of-the-art NMT models need much more computing power and data than SMT models, a requirement that is unsustainable in the long run and of very limited benefit in low-resource scenarios. To some extent, model compression—more specifically state-of-the-art knowledge distillation techniques—can remedy this. In this work, we investigate knowledge distillation on a simulated low-resource German-to-English translation task. We show that sequence-level knowledge distillation can be used to train small student models on knowledge distilled from large teacher models. Part of this work examines the influence of hyperparameter tuning on model performance when lowering the number of Transformer heads or limiting the vocabulary size. Interestingly, the accuracy of these student models is higher than that of the teachers in some cases even though the student model training times are shorter in some cases. In a novel contribution, we demonstrate for a specific MT service provider that in the post-deployment phase, distilled student models can reduce emissions, as well as cost purely in monetary terms, by almost 50%.

Keywords: NMT; Green AI; knowledge distillation; CO_2 savings

check for updates

Citation: Jooste, W.; Haque, R.; Way, A. Knowledge Distillation: A Method for Making Neural Machine Translation More Efficient. *Information* **2022**, *13*, 88. https://doi.org/10.3390/info13020088

Academic Editor: Kostas Stefanidis

Received: 3 December 2021
Accepted: 7 February 2022
Published: 14 February 2022

1. Introduction

Deep neural networks (DNN) underpin state-of-the-art applications of artificial intelligence (AI) in almost all fields, such as image, speech and natural language processing (NLP). However, DNN architectures [1] are often data-, compute-, space-, power- and energy-hungry, typically requiring powerful graphic processing units (GPUs) or large-scale clusters to train and deploy, which has been viewed as a "non-green" technology [2].

As a result of the European Green Deal https://ec.europa.eu/info/strategy/priorities-2019-2024/european-green-deal_en (accessed on 2 December 2021) and the Horizon Europe Work Programme for 2021–2022 adopted on 15 June 2021, the European Commission has committed to making Europe the world's first climate-neutral continent by 2050. If this important goal is to be achieved, more efficient AI models have to play their part in helping to reduce the amounts of energy that are required for data storage and algorithm training.

The concept of 'green labs' in the natural sciences is nowadays quite well-known https://www.mygreenlab.org/ (accessed on 2 December 2021). Ultimately, researchers could no longer avoid the issue, given the amount of plastic waste right in front of their eyes in the lab at the end of any given day. In contrast, in computing, the GPUs on which we build our models and run our experiments are hidden away in the cloud or in a refrigerated machine room somewhere out of sight. Accordingly, most AI practitioners fail to consider the amount of electricity consumed and CO_2 generated by the machine learning models that they (we!) build. That is starting to change https://datacenters.lbl.gov/ (accessed on 2 December 2021) but much more can—and needs to—be done.

In MT, shared tasks focusing on efficiency http://www.statmt.org/wmt21/efficiency-task.html (accessed on 2 December 2021) are helping in this regard. A recent paper by Yusuf et al. [3] tracks the energy consumption of training translation models across different language pairs, but this was met with a somewhat mixed reaction on Twitter, but at least it turned the topic into a discussion point and provoked a response.

While the current paper concentrates on how smaller, greener models of MT might be built, the seemingly inexorable drive towards larger DNNs has received attention in the area of image classification, where Thompson et al. [4] gathered data from more than a thousand research papers on deep learning and discussed their findings in detail. After analysing the data they found that, in practice, in order to halve the error rate, approximately 500 times the amount of resources used nowadays are required. If the gains obtained in recent years continue, by 2025 the error level in the best AI systems designed for recognising objects in the ImageNet [5] dataset might be reduced to just 5%. Figure 1 shows the infeasible amount of computing resources required to achieve such a small error rate, and the concomitant amount of CO_2 emitted. In the same vein, Table 1 shows the estimated CO_2 emissions from training common NLP models as calculated by Strubell et al. [2], compared to how much CO_2 is emitted over the lifetime of the average human, the average American, and a typical car.

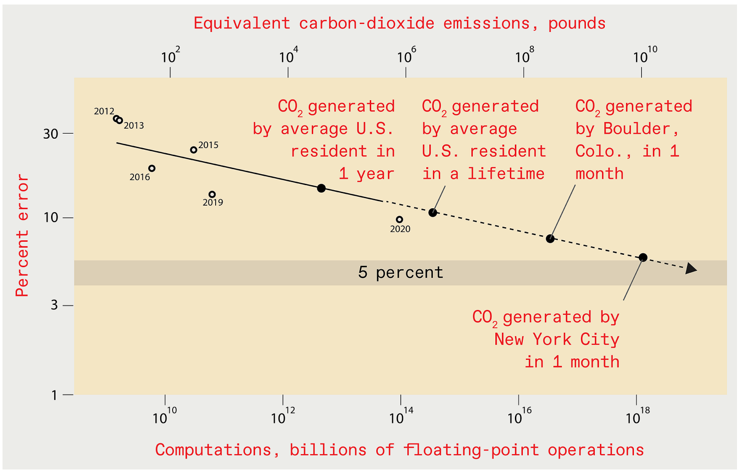

Figure 1. Comparing the computing resources and energy required to train a DNN system to recognise objects in the ImageNet dataset with an error rate of 5%. The amount of CO_2 emitted would be as much as New York City generates in one month. Figure courtesy of Thompson et al. [4].

Table 1. Estimated CO_2 emissions from training common NLP models, compared to familiar consumption. Table courtesy of Strubell et al. [2], with original sources for air travel and per-capita consumption: https://bit.ly/2Hw0xWc (accessed on 2 December 2021), and car lifetime: https://bit.ly/2Qbr0w1 (accessed on 2 December 2021).

Consumption	CO_2e (lbs)
Human life, avg, 1 year	11,023
American life, avg, 1 year	36,156
Car, avg incl. fuel, 1 lifetime	126,000
NLP pipeline (parsing, SRL)	39
w/ tuning and experimentation	78,468
Transformer (big)	192
w/ neural architecture search	626,155

The main idea behind model compression is to "compress" an ensemble of large models into a smaller model with minimal performance loss. This is generally done by using a small, fast model to approximate the function learned by a much larger and slower model with better performance [6]. Hinton et al. [7] show that compressing the knowledge from a cumbersome model into a smaller model can be seen as a mapping from input vectors to output vectors, and the relative probabilities of incorrect outputs can provide insight into how the cumbersome model tends to generalise. The work mentioned so far mainly investigated non-recurrent models used for classification tasks.

Knowledge distillation [6] can be used to transfer the knowledge from a teacher network (a large, slow model) to a student network (a small, fast model). This is a promising technique to disrupt the current situation for NLP tasks where almost all systems tend to use cumbersome DNN architectures.

The methods described by Bucilua et al. [6] and Hinton et al. [7] can be used for word-level knowledge distillation, since NMT models make use of multi-class prediction at the word-level. These models, however, need to predict complete sequences that are dependent on previous predictions as well.

Kim and Rush [8] proposed sequence-level knowledge distillation, where a new training set is generated by translating a dataset with the teacher model using beam search. The newly generated training set is then used to train a smaller student model. They show how the usual training criteria for multi-class classifiers can be used to develop a function for knowledge distillation, which can be expanded even further to be used for word-level knowledge distillation and finally sequence-level knowledge distillation.

Assume we want to classify the data in the set (x, y) into a set of classes V. The aim is to minimise the cross-entropy between the data distribution and model distribution p parameterised by θ. This can be done by minimising the negative log-likelihood (NLL) for each training example, as in (1):

$$\mathcal{L}_{NLL}(\theta) = -\sum_{k=1}^{|V|} \mathbb{1}\{y = k\} \log p(y = k|x;\theta), \tag{1}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function. In terms of knowledge distillation, we have a model distribution $q(y = k|x, \theta_T)$, learned by the teacher, so Equation (1) can be rewritten as (2):

$$\mathcal{L}_{KD}(\theta, \theta_T) = -\sum_{k=1}^{|V|} q(y = k|x, \theta_T) \log p(y = k|x;\theta). \tag{2}$$

We can now use $\mathcal{L}_{KD}$ to define functions for knowledge distillation for NMT. First, standard knowledge distillation can be applied to NMT models since word NLL is minimised during training. The standard function becomes (3):

$$\mathcal{L}_{WORD-KD}(\theta, \theta_T) = -\sum_{j=1}^{J}\sum_{k=1}^{|V|} q(t_j = k|s, t_{<j}) \log p(y = k|s, t_{<j}), \tag{3}$$

where V is the target vocabulary and t and s the target and source sentences, respectively. Finally, a loss function for sequence-level knowledge will be derived, since word-level knowledge distillations can easily lead to the forward propagation of incorrect predictions. Once again, we can use a probability distribution derived from the teacher model to define a loss function. Sequence distributions from the teacher model are used instead of word distributions and Equation (2) can thus be rewritten as (4):

$$\mathcal{L}_{SEQ-KD}(\theta, \theta_T) = -\sum_{t \in \mathcal{T}} q(t|s) \log p(t|s), \tag{4}$$

where $q(t|s)$ represents the sequence distribution over all possible sequences. This loss function, however, is complex to handle since it sums over an exponential number of terms. Kim and Rush [8] suggest the use of beam search to approximate Equation (4), which reduces the complexity of $\mathcal{L}_{SEQ-KD}$. It is worth noting that this method of knowledge distillation is difficult to apply when the domain of the training data is not well defined.

Currey et al. [9] introduce generalised sequence-level knowledge distillation to distil translations from domain-specific teacher models, and the knowledge distilled in their approach is then used to train a smaller, multi-domain student model. This approach is referred to as 'multi-domain knowledge distillation' (cf. Figure 2). It is worth noting that this method of knowledge distillation is difficult to apply when the domain of the training data is not well defined.

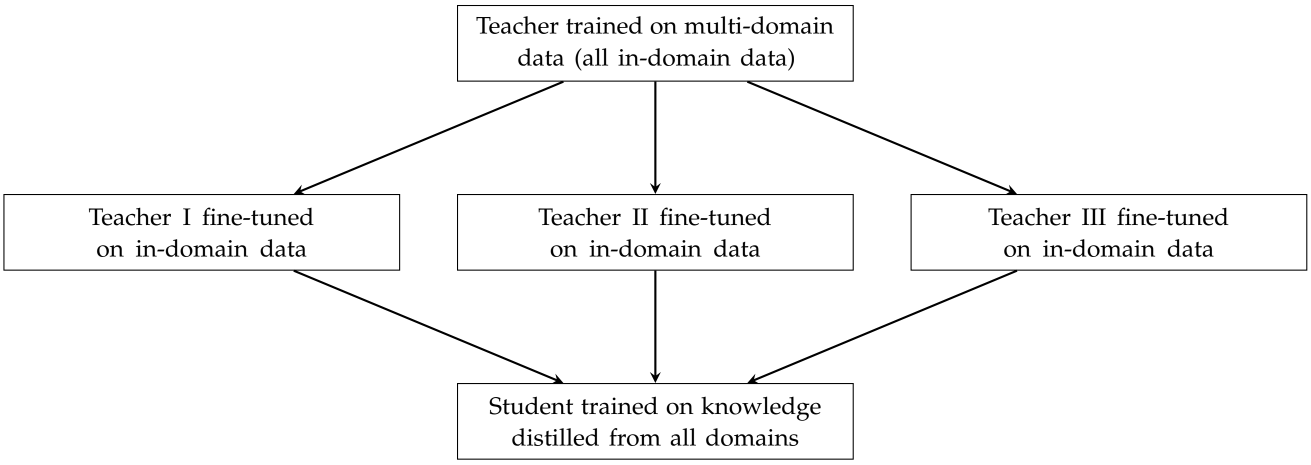
Teacher trained on multi-domain data (all in-domain data)
Teacher I fine-tuned on in-domain data
Teacher II fine-tuned on in-domain data
Teacher III fine-tuned on in-domain data
Student trained on knowledge distilled from all domains

Figure 2. Flowchart of the process of distilling multiple domains as in Currey et al. [9] to create a general domain student model.

Both Currey et al. [9] and Gordon and Duh [10] use similar architectures for their models, that is, teacher models with 12 encoder and decoder layers and student models with six encoder and decoder layers. Training teacher models with this type of architecture requires a large amount of memory and GPUs.

Zhang et al. [11] propose an adapted method of sequence-level knowledge distillation named 'dual knowledge distillation'. This method utilises bidirectional translation models to significantly improve translation in both directions.

Wang et al. [12] propose two strategies to select distilled knowledge for training student models, namely batch-level selection and global-level selection. The authors show the impact that different words and sentences have as carriers of knowledge, and how consistent improvements on various datasets can be achieved using these strategies.

Passban et al. [13] describe a new approach called 'attention-based layer projection' for knowledge distillation. In this approach, the output of each layer of the student model is compared to that of the teacher model, in order to help the student to produce better outputs.

Dakwale [14] uses knowledge distillation to address the problem of catastrophic degradation during domain adaptation. This was applied using an in-house NMT system rather than a recurrent neural network [15].

We take inspiration from this body of work, and summarise the main contributions of this paper as follows:

- We use sequence-level knowledge distillation and show that small student models can outperform large teacher models;
- We show that small student models prove to be very useful in the case where MT models need to be deployed in environments where constraining the available hardware is important;
- We demonstrate a translation industry scenario where knowledge distillation in NMT is used for translating sentences in large-scale projects. For a real, current provider, we focus on three parameters of translation projects which are of crucial importance in industrial settings, namely translation time, translation cost, and carbon emissions, and demonstrate that savings of almost 50% can be achieved;
- As our investigation focuses on the performance evaluation of small and large NMT models in a low-resource set-up, the findings in this paper add value to the current research on sustainable MT development;

- Our research provides an alternative, realistic solution to SMEs who are currently unable to provide MT solutions to their clients due to the huge deployment costs associated with large-scale NMT models;
- Our findings help to demonstrate that the sort of energy reductions required to achieve climate neutrality may be achieved.

When discussing our results in terms of the carbon emissions generated by these models, we use the framework of Henderson et al. [16] for tracking energy consumption and carbon emissions.

2. Experimental Setup

We use the Europarl https://opus.nlpl.eu/Europarl-v3.php (accessed on 2 December 2021) [17] corpus with parallel sentences in German and English for our NMT simulation experiments described in this section for the language direction German to English. The corpus is randomly divided into three subsets, namely the training set, validation set and test set. The training set consists of roughly 2 million sentences and the validation and test sets of 3000 sentences, respectively.

As for the preprocessing of the data, the Moses [18] toolkit was used to tokenize and clean the three datasets mentioned above by removing all sentences with a length greater than 100. The toolkit was also used to decase all sentences before training and after training, we used a pretrained truecaser to recase all translated sentences. Furthermore, SubwordNMT https://github.com/rsennrich/subword-nmt (accessed on 2 December 2021) was used to segment the sentences in the corpus into subword units as described by Sennrich et al. [19]. More specifically, the Byte Pair Encoding (BPE) vocabularies were set to 32*k* words.

The performance of all our models was measured with three evaluation metrics, namely BLEU [20], TER [21] and chrF https://github.com/m-popovic/chrF (accessed on 2 December 2021) [22], using the MultEval toolkit [23] https://github.com/jhclark/multeval (accessed on 2 December 2021). Of course, these metrics provide an indication of the quality of the translations produced by our NMT systems, but do not provide insight into the efficiency of our systems in terms of model size, number of parameters and training times.

The training set size, number of GPUs, time taken, electricity consumption and CO_2 production and the balance between these constraints and performance need to be taken into account if we are to report on the efficiency of our systems. Electricity consumption and CO_2 emissions can be estimated by taking training time and GPU specifications into account. In addition, human evaluation methods can provide better insight into the optimal balance of these factors, as automatic evaluation methods do not give an accurate indication of any deterioration in quality seen when smaller models are used. By the same token, continuing to train our DNN models for further epochs may result in gains according to automatic metrics which are not discernible to humans.

We use the MarianNMT https://github.com/marian-nmt/marian (accessed on 2 December 2021) toolkit [24] and Transformer [25] architecture to train the models for our experiments. All models were trained for a maximum of 20 epochs, since that was the lowest number of epochs needed to finish training for one of our models. Listing 1 shows an example training setup with most notable parameters for one of our baseline models.

The same script is used to train the student models, where the only difference is the training datasets used for training. The baseline and student models have the same architectures, since we want to determine the impact of the knowledge distilled from the teacher models on the efficiency of our systems.

Furthermore, we experimented with the hyperparameters of the student models by training these models with four Transformer heads, compared to eight heads, as shown in line 6 of Listing 1. We also tested the performance of our student models by limiting the vocabulary sizes to 8*k* and 16*k* tokens. The results of this set of experiments are presented in Section 3.4.

Listing 1. Baseline model parameters.

```
--mini-batch-fit -w 9000 --mini-batch 1000 --maxi-batch 1000 \
--valid-mini-batch 64 \
--cost-type=ce-mean-words \
--beam-size 12 --normalize 1 \
--enc-depth 3 --dec-depth 3 \
--transformer-heads 8 \
--transformer-postprocess-emb d \
--transformer-postprocess dan \
--transformer-dropout 0.1 --label-smoothing 0.1 \
--learn-rate 0.0003 \
--lr-warmup 16000 --lr-decay-inv-sqrt 16000 --lr-report \
--optimizer-params 0.9 0.98 1e-09 --clip-norm 5 \
--tied-embeddings --exponential-smoothing
```

As for the teacher models, the `--enc-depth` and `--dec-depth` parameters were set to 6, instead of 3. Other than the difference in encoder and decoder layers, the script remains the same and the teacher models are trained on the same training sets as the baseline models. Table 2 summarises the three types of models that were used in our experiments and changes in their architectures.

Table 2. Comparison of baseline, teacher and student models. (KD is the knowledge-distilled training set).

Model Type	Corpus	# of enc/dec Layers
Baseline	EuroParl	3
Teacher	EuroParl	6
Student I	KD	3
Student II	KD + EuroParl	3

3. Results

3.1. Baseline Models

All three baseline models have the same parameters as in the example shown above. The difference between these models relates to the number of GPUs used during training time. Baseline 1 was trained using one GPU, Baseline 2 using two GPUs and Baseline 3 using four GPUs. For each of these models, the training time and translation accuracy is compared in Table 3. As expected, the training time goes down as the number of GPUs increases. Interestingly, Baseline-2GPU has the best BLEU score when training is limited to 20 epochs, although the differences in the scores of all the baseline models are not significant. After further investigation, the reason behind Baseline-2GPU performing better than Baseline-4GPU has to do with the way in which the MarianNMT toolkit saves model weights and run-time parameters.

For our experiments the save- and validation-frequency is set to 5000 iterations, whereas for Baseline-2GPU the model was saved during epoch 19 (iteration 50,000), and Baseline-4GPU was saved during epoch 18 (iteration 25,000). Iteration 50, 000 only occurred during epoch 36. Further investigation needs to be done on the method in which these iterations are saved in order to change these parameters so they are more efficient; in Section 4 where we discuss Table 2, note that the same behaviour ensues.

Table 3. Comparison of the performance of the baseline models.

Baseline	# of GPUs	Training Time	BLEU	TER
1	1	07:50:13	26.39	48.3
2	2	05:07:48	26.68	48.2
3	4	03:50:52	26.39	48.4

3.2. Teacher Models

Three teacher models were also trained using the Europarl corpus and the number of GPUs used during training was the same as for the baseline models. However, these models have six encoder and decoder layers, which contrasts with the baseline and student models which have only three encoder and decoder layers. The results for the teacher models are shown in Table 4. Once again, the difference in accuracy is not very significant across all the metrics. The difference in training time of Teacher-GPU2 and Teacher-GPU4 is only 20 min, so in our future work we aim to track the energy usage of 2 GPUs compared to 4 GPUs in order to see which setup leaves a smaller carbon footprint. This is especially important given the only very slight improvements in translation quality, albeit demonstrated by all three metrics.

Table 4. Comparison of the performance of the teacher models.

Teacher	# of GPUs	Training Time	BLEU	TER
1	1	13:25:46	26.60	48.1
2	2	08:05:38	26.71	47.7
3	4	07:48:01	26.76	47.5

3.3. Student Models

Similar to the baseline models, the student models have just three encoder and decoder layers. However, the student models are trained on new training sets created by translating the German sentences in the training set into English using each teacher model with a beam size of 12. The original German sentences and the newly translated English sentences were then used to create the new parallel training corpus. This corpus is referred to as the 'KD training set'.

The first three student models were trained on the KD set only, and the following three student models were trained on the original training set appended to the KD set [26]. Each student model's training set has a KD set that was translated by a teacher model with a corresponding number of GPUs used during training. The training sets used, number of GPUs used, training time and evaluation scores are shown in Table 5.

Table 5. Comparison of the performance of the student models.

Student	Training Set	# of GPUs	Training Time	BLEU	TER	chrF
1	KD	1	07:07:56	26.66	49.5	59.35
2	KD	2	04:46:34	26.49	49.3	59.51
3	KD	4	04:21:48	26.22	50.1	59.11
4	EuroParl+KD	1	16:48:06	26.68	48.8	59.58
5	EuroParl+KD	2	10:25:41	26.87	48.6	60.11
6	EuroParl+KD	4	08:44:18	26.81	49.0	59.85

3.4. Hyperparameter Tuning

The performance of the experimental student models is shown in Tables 6 and 7. While it is interesting to see the exact scores for all set-ups, in order to help the reader more easily see the overall picture, we encapsulate this information in Figures 3–5. Figure 3 presents the average accuracy of the student models when limiting the vocabulary sizes at training. Since the training sets and number of GPUs used are consistent for student models 1 to

6, they are not shown. In Figure 4, we see the performance of the student models when the number of Transformer heads is reduced from 8 to 4. The vocabulary size was left at 32*k*. Once again the training sets and number of GPUs used are consistent. In future work, further experimentation will be done on combining smaller vocabulary sizes with a smaller number of Transformer heads. Figure 5 shows the average training time comparison of the various models. We discuss all these results in the next section.

Table 6. Comparison of the performance of the student models with varying vocabulary sizes.

Student	Vocabulary Size	Training Time	BLEU	TER	chrF
1	8*k*	05:10:08	22.63	55.63	52.44
2	8*k*	02:49:39	22.83	55.21	52.95
3	8*k*	03:05:43	21.72	57.06	51.23
4	8*k*	11:03:00	23.55	54.03	53.79
5	8*k*	06:00:09	23.41	53.61	54.04
6	8*k*	06:07:55	23.17	54.14	53.46
1	16*k*	06:08:42	25.30	51.42	57.10
2	16*k*	03:22:43	25.23	51.29	57.28
3	16*k*	03:18:20	24.36	52.87	56.01
4	16*k*	13:03:38	25.93	50.59	57.89
5	16*k*	07:08:26	25.74	50.41	57.89
6	16*k*	07:22:54	25.73	50.83	57.58

Table 7. Performance of the student models with 4 Transformer heads.

Student	Transformer Heads	Training Time	BLEU	TER	chrF
1	4	06:46:49	26.27	50.10	58.93
2	4	04:55:52	26.52	49.56	59.37
3	4	03:11:03	25.92	50.61	58.64
4	4	15:32:27	26.69	49.04	59.65
5	4	10:15:02	26.90	48.49	60.14
6	4	07:42:52	26.83	49.05	59.72

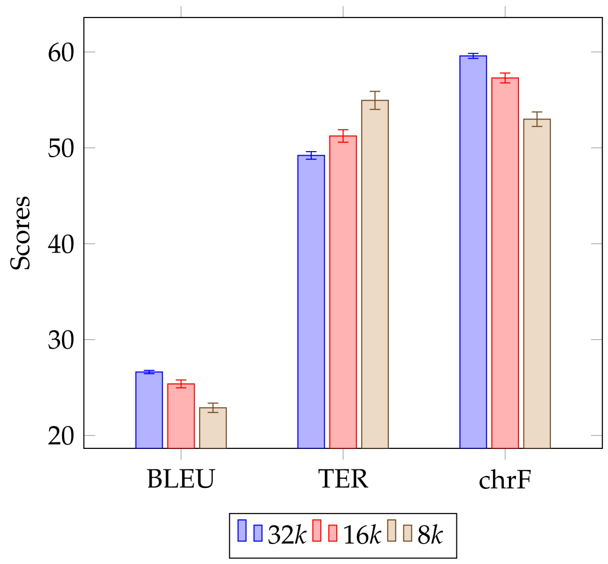

Figure 3. Average score comparison of student models with varying vocabulary sizes, including the standard deviations. Note that BLEU and chrF are precision-based metrics, so the higher the score the better, whereas TER is an error metric, so lower scores indicate better quality.

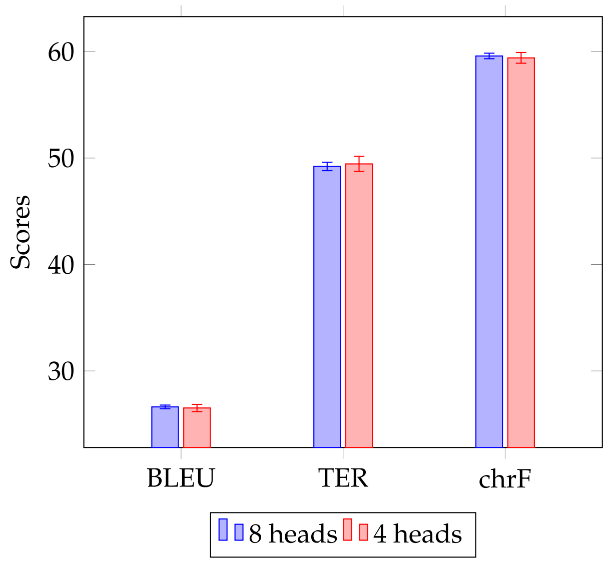

Figure 4. Average score comparison of student models with varying numbers of Transformer heads, including the standard deviations.

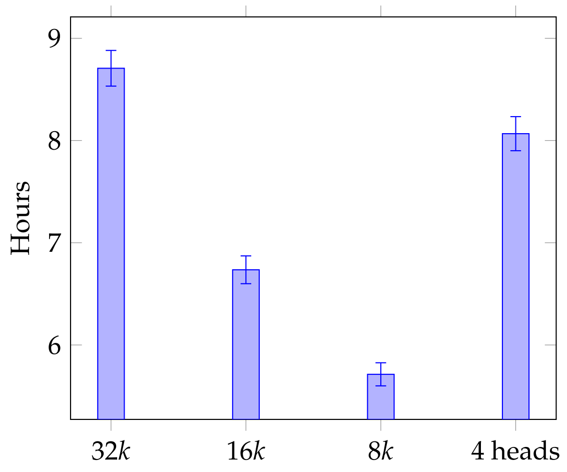

Figure 5. Average time comparison of student models with varying vocabulary sizes and Transformer heads. Where the vocabularies are specified, the number of Transformer heads is equal to 8, while for 4 Transformer heads the vocabulary size is set to 32*k*.

4. Discussion

When comparing the results of the baseline models to the results of the teacher models, as can be seen from Tables 3 and 4, the teacher models perform better than the baseline models as far as all three automatic metrics are concerned. When the vocabulary size is varied, as shown in Figure 3, the performance of the student models decreases consistently with each halving of vocabulary size, for all three evaluation metrics. However, lowering the number of Transformer heads does not seem to significantly affect quality, as Figure 4 shows.

In terms of the training times for the baseline models compared to the teacher models, the scores improve marginally, but the training times are longer than those of the baseline models. In fact, the best scores are obtained with Teacher-4GPU. Figure 5 shows the average training time for the various student models and it is clear that the models with smaller vocabularies (16*k* and 8*k*) trained faster on average, albeit with lower quality in terms of runtime performance as we have just seen. In comparison, the models with 4 Transformer heads (and a vocabulary size of 32*k*) trained slightly faster than the original student models, and with no significant deterioration in quality.

As for Baseline-4GPU and Teacher-4GPU, we compared translations and observed that even though there is a statistically significant difference in BLEU scores (calculated via approximate randomisation using the MultEval tool), the actual translations produced by both models do not differ significantly in quality from a human perspective. Example translations produced by the MT systems are shown in Table 8, where from a human point of view, the translations produced by the two systems are equally valid. Accordingly, it appears that training a model with only three encoder and decoder layers is justifiable in the case of limited computing resources, since the sentences in both cases remain accurate and fluent, and no worse than a model using twice the amount of layers.

Table 8. Comparison of original English reference translations to MT outputs.

Source	Sentence
EuroParl	And with this extension, when the decisions are legislative, will there always be codecision with Parliamemnt?
Baseline-4GPU	And when it comes to legislative decisions, will Parliament always have codecision?
Teacher-4GPU	And will Parliament always have codecision when it comes to legislative decisions?
EuroParl	Almost all the speakers in this debate have mentioned credit rating agencies.
Baseline-4GPU	Almost every speaker spoke today about credit rating agencies.
Teacher-4GPU	Almost every speaker has spoken today about credit rating agencies.

If we compare Student-KD and Teacher models, training time is almost half the amount of time for all Student models. For Student-KD-1GPU, the BLEU score is nearly identical to that of the corresponding Teacher model, but the TER and chrF scores show it to be a little worse. If we compare the Student-KD models to the Student-KD+EP models, the latter takes longer to train and leads to better scores for all automatic metrics.

From Figure 3, it is clear that limiting the size of the vocabulary causes the experimental student models to train faster than the original student models (by up to 2 h). The accuracy, however, is lower (by about 2.5 BLEU points, or 10% relative), especially for models trained only on the KD set. The anomaly when using 2 GPUs appears in Table 6 for both vocabulary sizes too, and it again occurs when the model iteration is not saved during the last epoch.

Interestingly, the model with the best accuracy was trained on the EuroParl+KD training set using 2 GPUs and only 4 Transformer heads. In some cases the student models with 4 Transformer heads were slightly quicker to train and more accurate than the original student models with 8 Transformer heads. Figure 4 shows that the accuracy is on average very similar when using the different number of Transformer heads during training, but the average training time for models trained using only 4 Transformer heads is 7 h less than when using 8 Transformer heads.

Kim and Rush [8] showed that when using a long short-term memory architecture, some smaller student MT models outperformed large teacher models. This is also the case for our Transformer models when using the original (Europarl) and KD training set as shown in Table 9.

Table 9. Difference in BLEU scores between teacher and student models.

# of GPUs	Teacher	Student-KD+EP	Difference
1	26.60	26.68	+0.08
2	26.71	26.87	+0.16
4	26.76	26.81	+0.05

Impact

In this section, we consider a number of matters which are important industry concerns in the post-deployment phase, where only the translation process itself is taken into account and not the preceding training process. To the best of our knowledge, this study is the first of its type to take a realistic scenario from an actual service provider and estimate the savings that can be achieved from distilling larger models into small ones.

Table 10 shows the time (in seconds) it took for our models to translate the test set we used for evaluation, as described in Section 2. Figure 6 shows some of the site statistics available on the KantanMT Platform https://www.kantanmt.com/ (accessed on 2 December 2021). It is clear from Table 10 that the student models translate the source sentences much faster than the teacher models. Somewhat more interestingly, we observe that using 2 GPUs for translation yields the fastest translation time; more specifically, model Student-KD+EP-2GPU takes 29.49 s to translate 3000 sentences. Furthermore, there are 69,543 German words in the test file that was translated, so model Student-KD+EP-2GPU translated on average 2358 words per second.

Table 10. Comparison of time taken to translate the test set in seconds.

# of GPUs	Teacher	Student-KD+EP
1	84.10	45.20
2	54.09	29.49
4	69.80	37.05

Taking the statistics in Figure 6 as an example of where MT is used in the translation industry, an average of 255,546,342 words are translated per month. This average is denoted as M_{avg}. We based our calculations of the cost and carbon emissions on using an NMT model to translate this average amount of words per month. These results are only estimates, and in the future we aim to track these variables more precisely, but they are quite insightful nonetheless. Schwartz et al. [27] point out that tracking these variables is very dependable on the deployment environment since carbon emissions are highly dependable both on the local electricity infrastructure and the type of hardware that is used. Thus, we will need to repeat these experiments in a controlled environment in future work.

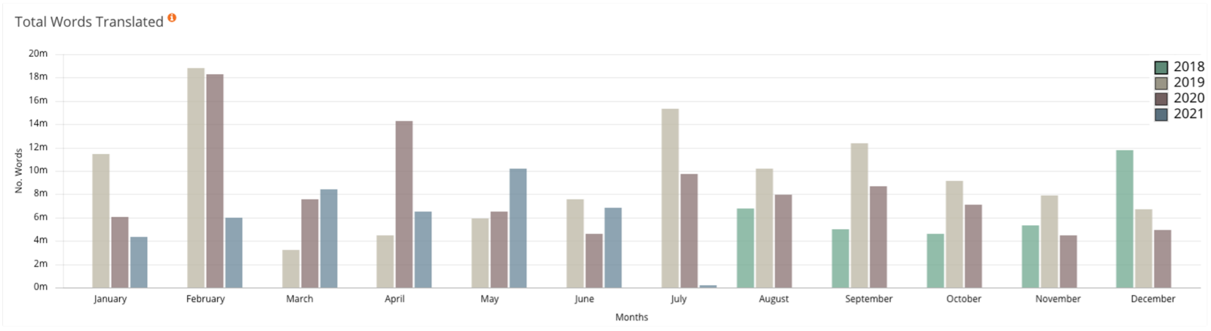

Figure 6. Total number of words translated per month on the KantanMT Platform. The statistics provided are from 1 August 2018 and are updated every 2 h.

Estimates of the carbon emissions were conducted using the Machine Learning Impact Calculator presented by Lacoste et al. [28].

As for the cost, we used the AWS Pricing Calculator. The AWS Pricing Calculator (https://calculator.aws/#/) (accessed on 2 December 2021) provides only an estimate of one's AWS fees to estimate the cost of using AWS GPUs for a given amount of hours per month. We use the *p3.8xlarge* https://aws.amazon.com/ec2/instance-types/p3/ (accessed on 2 December 2021) instance for pricing calculations which provides access to 4 NVIDIA https://www.nvidia.com/en-us/ (accessed on 2 December 2021) Tesla V100 GPUs, as well as 32 GB EBS Storage. The On-Demand Instances pricing is used.

In order to estimate a translation time for the cost and carbon emissions, we first calculated the average number of words translated per second by each model and then estimated the time it would take to translate M_{avg} words. These estimates are shown in Table 11. They show pretty clearly that the student models are much more efficient in terms of cost and CO_2 emissions when a model is deployed by industry to provide MT as a service. While the results are preliminary, and further investigation is required, we believe them to be encouraging, and a tentative endorsement of the role that distilled models can play in reducing the carbon footprint of the AI models that we build.

Table 11. Comparison of Cost in USD and CO_2 emissions of each model when translating M_{avg} amount of words per month. The Translation time in hours is an estimate only.

Model	M_{avg} (Translation Time)	Cost (USD)	CO_2 Emissions ($kgCO_2$-eq)
Teacher-1GPU	85.84	1,140.44	13.31
Student-KD+EP-1GPU	46.14	624.86	7.15
Teacher-2GPU	55.21	743.84	8.56
Student-KD+EP-2GPU	30.10	413.34	4.67
Teacher-4GPU	71.24	955.36	11.04
Student-KD+EP-4GPU	37.82	505.88	5.87

5. Conclusions

In this work, we showed that sequence-level knowledge distillation can be used to reduce model size and training time, without significant loss in performance; in some instances, it even leads to performance gains. It is also clear from our investigation that smaller models can be used when time and space constraints apply without a notable loss in performance. We also showed how changing hyperparameters can impact the training time and accuracy of the student models. Smaller vocabulary sizes lead to faster training times, but the accuracy of these models is quite a bit lower. Using only 4 Transformer heads caused training time to be only slightly faster but, interestingly, with little deterioration in terms of quality; again, in some cases the accuracy actually improved.

An important finding of this work is that student models are much more efficient in terms of cost and CO_2 emissions as far as offering MT as a service in the translation industry is concerned.

In the future, we plan to calculate the uncertainty score of each sentence translated by our teacher model. The idea is not to consider or undersample those sentences for which our teacher model is quite uncertain. Those sentences having low uncertainty scores can also be upsampled in the training sets of the student models so that the student models better mimic the characteristics of the teacher models.

In extensions to our hyperparameter experiments, we plan to combine the vocabulary size and Transformer head parameter modifications to obtain better insight into the impact of these changes. The size of these models also needs to be taken into account and compared to the original student and teacher models.

Furthermore, we aim to experiment on a bigger variety of training parameters when distilling knowledge from the teacher model. We also want to carry out these experiments on different domains and language-pairs, especially those in real resource-constrained environments as opposed to the simulations used here.

In follow-on work, we intend to use the framework of [16] in order to generate more accurate estimates of carbon emissions. This will be a crucial component of our overall aim to come up with a composite metric which takes all the parameters at play into account, so as to indicate what system configuration can best deliver on a client's wishlist for the lowest cost, while being as kind as possible to the environment.

Author Contributions: Writing—original draft, W.J.; Writing—review and editing, R.H. and A.W. All authors have read and agreed to the published version of the manuscript.

Funding: This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number 18/CRT/6183. The ADAPT Centre for Digital Content Technology (https://www.adaptcentre.ie/, accessed on 2 December 2021) is funded under the Science Foundation Ireland Research Centres Programme (Grant number 13/RC/2106) and is co-funded under the European Regional Development Fund. We also wish to thank KantanAI (https://www.kantanai.io/, accessed on 2 December 2021) for hosting the first-named author on her ML-Labs (https://www.ml-labs.ie/, accessed on 2 December 2021) placement, and for permission to use the statistics quoted in this paper.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
2. Strubell, E.; Ganesh, A.; McCallum, A. Energy and Policy Considerations for Deep Learning in NLP. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July 2019; pp. 3645–3650.
3. Yusuf, M.; Surana, P.; Gupta, G.; Ramesh, K. Curb Your Carbon Emissions: Benchmarking Carbon Emissions in Machine Translation. *arXiv* **2021**, arXiv:2109.12584.
4. Thompson, N.C.; Greenewald, K.; Lee, K.; Manso, G.F. Deep Learning's Diminishing Returns. 2021. Available online: https://spectrum.ieee.org/deep-learning-computational-cost (accessed on 2 December 2021)
5. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
6. Buciluǎ, C.; Caruana, R.; Niculescu-Mizil, A. Model compression. In Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, 20–23 August 2006; pp. 535–541.
7. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**, arXiv:1503.02531.
8. Kim, Y.; Rush, A.M. Sequence-Level Knowledge Distillation. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 1317–1327.
9. Currey, A.; Mathur, P.; Dinu, G. Distilling Multiple Domains for Neural Machine Translation. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 3 June 2020; pp. 4500–4511.
10. Gordon, M.; Duh, K. Distill, Adapt, Distill: Training Small, In-Domain Models for Neural Machine Translation. In Proceedings of the Fourth Workshop on Neural Generation and Translation, Online, 10 July 2020; pp. 110–118.
11. Zhang, H.; Qiu, S.; Wu, S. Dual knowledge distillation for bidirectional neural machine translation. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 15 December 2021; pp. 1–7.
12. Wang, F.; Yan, J.; Meng, F.; Zhou, J. Selective Knowledge Distillation for Neural Machine Translation. *arXiv* **2021**, arXiv:2105.12967.
13. Passban, P.; Wu, Y.; Rezagholizadeh, M.; Liu, Q. ALP-KD: Attention-Based Layer Projection for Knowledge Distillation. In Proceedings of the the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21), Online, 2–9 February 2021; pp. 13657–13665.
14. Dakwale, P. Strategies for Effective Utilization of Training Data for Machine Translation. Ph.D. Thesis, University of Amsterdam, Amsterdam, The Netherlands, 2020.
15. Mikolov, T.; Karafiát, M.; Burget, L.; Cernockỳ, J.; Khudanpur, S. Recurrent neural network based language model. In Proceedings of the 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, 26–30 September 2010; Volume 2, pp. 1045–1048.
16. Henderson, P.; Hu, J.; Romoff, J.; Brunskill, E.; Jurafsky, D.; Pineau, J. Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning. *J. Mach. Learn. Res.* **2020**, *248*, 1–43.
17. Tiedemann, J. Parallel Data, Tools and Interfaces in OPUS. In Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12), Istanbul, Turkey, 23–25 May 2012; pp. 2214–2218.
18. Koehn, P.; Hoang, H.; Birch, A.; Callison-Burch, C.; Federico, M.; Bertoldi, N.; Cowan, B.; Shen, W.; Moran, C.; Zens, R.; et al. Moses: Open Source Toolkit for Statistical Machine Translation. In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions, Prague, Czech Republic, 25–27 June 2007; pp. 177–180.

19. Sennrich, R.; Haddow, B.; Birch, A. Neural Machine Translation of Rare Words with Subword Units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016; pp. 1715–1725.
20. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. BLEU: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, Philadelphia, PA, USA, 7–12 July 2002; pp. 311–318.
21. Snover, M.; Dorr, B.; Schwartz, R.; Micciulla, L.; Makhoul, J. A study of translation edit rate with targeted human annotation. In Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers, Cambridge, MA, USA, 8–12 August 2006; pp. 223–231.
22. Popović, M. chrF: Character n-gram F-score for automatic MT evaluation. In Proceedings of the Tenth Workshop on Statistical Machine Translation, Lisbon, Portugal, 17–18 September 2015; pp. 392–395.
23. Clark, J.H.; Dyer, C.; Lavie, A.; Smith, N.A. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OG, USA, 19–24 June2011; pp. 176–181.
24. Junczys-Dowmunt, M.; Grundkiewicz, R.; Dwojak, T.; Hoang, H.; Heafield, K.; Neckermann, T.; Seide, F.; Germann, U.; Aji, A.F.; Bogoychev, N.; et al. Marian: Fast Neural Machine Translation in C++. In Proceedings of the ACL 2018, System Demonstrations, Melbourne, Australia, 15–20 July 2018; pp. 116–121.
25. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All you Need. In Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
26. Gordon, M.A.; Duh, K. Explaining Sequence-Level Knowledge Distillation as Data-Augmentation for Neural Machine Translation. *arXiv* **2019**, arXiv:1912.03334.
27. Schwartz, R.; Dodge, J.; Smith, N.A.; Etzioni, O. Green AI. *Commun. ACM* **2019**, *63*, 54–63. [CrossRef]
28. Lacoste, A.; Luccioni, A.; Schmidt, V.; Dandres, T. Quantifying the Carbon Emissions of Machine Learning. *arXiv* **2019**, arXiv:1910.09700.