Losing Heads in the Lottery: Pruning Transformer Attention in Neural Machine Translation

Maximiliana Behnke
The University of Edinburgh
maximiliana.behnke@ed.ac.uk

Kenneth Heafield
The University of Edinburgh
kheafiel@inf.ed.ac.uk

Abstract

The attention mechanism is the crucial component of the transformer architecture. Recent research shows that most attention heads are not confident in their decisions and can be pruned after training. However, removing them before training a model results in lower quality. In this paper, we apply the lottery ticket hypothesis to prune heads in the early stages of training, instead of doing so on a fully converged model. Our experiments on machine translation show that it is possible to remove up to three-quarters of all attention heads from a transformer-big model with an average -0.1 change in BLEU for Turkish→English. The pruned model is 1.5 times as fast at inference, albeit at the cost of longer training. The method is complementary to other approaches, such as teacher-student, with our English→German student losing 0.2 BLEU at 75% encoder attention sparsity.

1 Introduction

The transformer model (Vaswani et al., 2017) performs well for a variety of tasks, including neural machine translation (Dong et al., 2018; Grundkiewicz and Junczys-Dowmunt, 2019; Junczys-Dowmunt, 2018). However, like many neural networks, it is overparametrised, and inference is costly. Attention heads are the headline feature of the transformer model, essential to learning relationships between words as well as complex structural representations. Voita et al. (2019) showed that many of these heads could be pruned in a fully trained model, but removing the same heads before training yielded lower quality. We investigate a third way: pruning heads in early training. Empirically our method enables even more pruning, which is useful for faster machine translation.

Reinitialising a model with the same pruned structure underperformed in Voita et al. (2019), which is consistent with the lottery ticket hypothesis (Frankle and Carbin, 2019). According to the lottery ticket hypothesis, randomly initialising a model is akin to buying lottery tickets and a smaller network, such as a pruned model, buys fewer tickets. Prior lottery ticket research prunes individual parameters to form a sparse network; we show that this logic extends to entire transformer heads. We follow lottery ticket training strategies (Frankle et al., 2019) to prune in early training, achieving a better trade-off between pruning and quality than pruning after training (Voita et al., 2019).

Our main goal is faster inference speed for machine translation deployment with minimal impact on quality. Pruning heads means they can be removed from the model entirely (with other heads shifted down), resulting in a layer configured to have fewer heads. Unlike most work on pruning (Zhu and Gupta, 2017; Gale et al., 2019), there is no need for sparse matrices, block-sparse matrix operators, or additional masking. In particular, we go further than Voita et al. (2019) by removing rather than masking.

In this paper we combine findings of both Voita et al. (2019) ("what") and Frankle and Carbin (2019) ("how") to prune attention heads. First, we define a training scheme based on an iterative approach that does not require full convergence of a model each time partial pruning takes place. To analyse the impact of pruning in a variety of settings, we experiment with a stock and highly optimised system across two language pairs: Turkish→English and English→German. We present and analyse our results in Sections 7 and 8.

Our key findings show that:

1. The lottery ticket hypothesis can be applied to prune whole blocks of parameters, instead of removing them separately.

2. Most attention heads can be removed early into training without significant damage to quality. The most aggressive attention pruning loses about 1 BLEU point with 80-90% block sparsity.

3. The lottery ticket approach achieves better results than a model trained from scratch with the same structure.

4. Pruned models exhibit patterns in regards to a number of heads in layers. For example, context attention gets more important as layers go on. The decoder requires self-attention only in the first layer — the rest is redundant, thus removable.

2 Related work

Magnitude pruning is one of the simplest algorithms, in which the smallest weights are removed. Successfully applied to NMT (See et al., 2016), this method works on a coefficient level and often requires retraining to recover the damage done by pruning. Further research shows that training a model from scratch with the same structure as the pruned one yields subpar results. Finishing training is a necessary step to reduce the size of a model (Gale et al., 2019) without too much damage. However, the sparsity of singular weights is generally too low to be efficiently exploited by a CPU or GPU. Block sparsity (Narang et al., 2017) is more hardware friendly because masked blocks can be skipped entirely. In this paper, we concentrate on a specific case of block sparsity that removes entire attention heads from a model without masking.

Brix et al. (2020) applied the lottery ticket hypothesis and other techniques to prune individual coefficients from a transformer for machine translation. In their experiments, a stabilised version of lottery ticket pruning damages translation quality by 2 BLEU points while removing 80% of all parameters. They improve upon that further by proposing a mix of lottery ticket and magnitude pruning. In their work, all layers are pruned the same amount, whereas our work prunes globally to reveal which layers can be pruned more aggressively. They aimed to compress the model and did not report any speed results, subsequently clarifying after their presentation that they did not achieve a speed improvement. Here, we aim for speed and only marginal improvements to size. Rather than prune individual coefficients, we pruned entire heads which can then be removed from the model entirely without even calling a sparse matrix routine.

Pruning is usually done at the end of training and then requires either retraining or tuning. There is an ongoing research field on integrating pruning into training For example, Golub et al. (2018) pruned weights that have accumulated the lowest total gradients and reduces the memory footprint to allow training much larger models than possible on available hardware. Our lottery ticket method does not require to modify a training algorithm and can be easily scripted to work "out of the box" with existing toolkits.

Xiao et al. (2019) observed that numerous computations in the attention mechanism are redundant with many layers sharing similar distributions. They proposed reusing attention output within adjacent layers in a model, which requires a model to learn which layers should be allowed to share outputs. This reuse of parameters could be understood as a pruning method that concentrates on removing vertical redundancy, in contrast to our research, which is more horizontal.

Since the attention mechanism is expensive to use in a decoder — with $O(n^2)$ complexity looping when generating translations — the better option would be to replace it with less expensive equivalent. In our teacher-student experiments, Simpler Simple Recurrent Unit (SSRU) (Kim et al., 2019) replaces the decoder self-attention mechanism. Still, this approach leaves an encoder and context between them unchanged. The lottery ticket pruning is complementary and can remove encoder and context heads on the top of it.

Looking into an impact of attention on output, Serrano and Smith (2019) analysed a text classification task whether "high attention weights correlate with greater impact on model predictions". They argued that, in contrast to a simple classification, "for tasks with a much larger output space (such as language modelling or machine translation) . . . almost anything may flip the decision". However, according to our experiments, careful head removal based on their importance does not damage quality.

3 Background

The usual approach to pruning assumes that a model is converged first and pruned second, optionally with continued training. Frankle et al.

(2019) have shown that iteratively pruning a model uncovers smaller and better quality subnetworks in comparison to pruning just once at the end. Still, training a model until convergence at every pruning iteration is too expensive to utilise for most architectures. For this reason, Frankle et al. (2019) introduced *late resetting* and *early turnaround*. Both of these methods combined shorten training time of each step in the iterative lottery scheme. Late resetting reverts parameters after pruning back to a checkpoint from early stages of training, not to the starting initialisation. Early turnaround means a model does not need to be fully trained to make a pruning decision but can approximate that by doing short training loops.

Lottery ticket pruning has been applied to natural language processing (NLP) tasks, including NMT (Yu et al., 2020). The winning ticket for that task was "remarkably robust to pruning" of singular weights if embeddings were spared from pruning. However, Yu et al. (2020) noted a linear drop in BLEU with sparsit.

Voita et al. (2019) analysed the attention mechanism and noticed that the majority of heads are useless: they either do not have linguistically interpretable roles or cannot make reliable choices when making alignments. Those heads were pruned by tuning a model with a L_0 regulariser that progressively switched off less essential heads. The L_0 regulariser needs a model to be fully trained first and then pruned while tuned. In contrast, our paper focuses on pruning heads as early as possible in training so that a model can converge with them removed. Using their selection heuristic, empirically we can safely prune more heads overall.

4 Methodology

In this section, we describe the lottery ticket approach as well as the decision heuristic based on attention importance (Voita et al., 2019) to remove heads in our models.

4.1 Lottery ticket

We apply an iterative pruning strategy based on Frankle and Carbin (2019), which introduced the lottery ticket hypothesis:

> *A randomly-initialized, dense neural network contains a subnetwork that is initialized such that – when trained in isolation – it can match the test accuracy of the original network after training for at most the same number of iterations.*

In other words, some parts of the network were luckily initialized and perform most of the work.

One could train a complete model, identify unlucky heads with a pruning heuristic, and retrain the pruned model starting with the same initialization.[1] This approach is expensive because the model is trained twice. Frankle et al. (2019) pointed out that unlucky parameters can be identified earlier in convergence, so it is not necessary to fully train a complete model first. We follow their work by partially training a model to make a pruning decision.

Frankle and Carbin (2019) reported that pruning iteratively yields smaller higher-quality networks that converge faster than those pruned in a single round. Removing most of the attention heads in one go seems too drastic using a simple heuristic, since other heads in layers may adapt to having fewer parameters and the roles of pruned heads may even transfer to those that are still active. For all these reasons, we apply a loop that iteratively prunes attention heads guided by partial training (Section 2). The training scheme is presented in Figure 1.

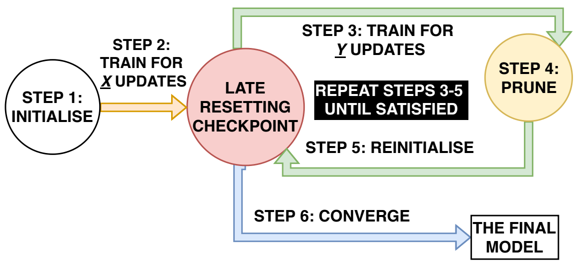
STEP 1: INITIALISE
STEP 2: TRAIN FOR X UPDATES
LATE RESETTING CHECKPOINT
STEP 3: TRAIN FOR Y UPDATES
REPEAT STEPS 3-5 UNTIL SATISFIED
STEP 4: PRUNE
STEP 5: REINITIALISE
STEP 6: CONVERGE
THE FINAL MODEL

Figure 1: Iterative lottery ticket pruning.

First, we train a model for a set number of updates and keep it as a late resetting checkpoint. Then the pruning phase starts — the model trains for a while, and selected heads are removed to have other parameters reinitialised to the checkpoint mentioned earlier at the end. That loop repeats until we are satisfied with how many parameters were removed. Finally, the pruned model can be converged.

[1]The order of data fed into a model is also retained among experiments.

4.2 Attention confidence

The lottery ticket hypothesis explains how pruning should progress, but the question remains: which heads should be removed in each pruning iteration? Inspired by Voita et al. (2019), we are mostly interested in heads that are *confident* in their decisions, which Voita et al. (2019) has shown to correlate with major identifiable roles attention heads performs. In their analysis, an attention head is defined as confident when it assigns a large weight to one of the words within a sentence That head should routinely make strong alignments to be considered a candidate to remain in a model.

When a head appears, its softmax layer computes a probability distribution over the words it attends to. We record the maximum of this probability distribution as confidence. For example, a context head attends over source words s.

$$\text{confidence} = \max_{s} \text{attention}(s)$$

These confidence values are averaged over all times the head appears while translating a development corpus. For example, a context head appears once per word in the output, so its confidence is averaged over all words in the output.

5 Baseline approaches

5.1 Just fewer attention heads

Do we even need to prune attention heads at all? Can we train a model that has fewer heads from the beginning? The typical transformer implementation described by Vaswani et al. (2017) initialises attention matrices based on the embedding dimension and those matrices are split into separate heads. That means the fewer heads there are set to be in a model, the larger they are. To compare models with different number of heads fairly, we fix their size to a constant instead.

We use all the parallel data allowed by the constrained condition of the WMT17 news task (Bojar et al., 2017) for English→German ($4.56M$ sentences) following a standard preprocessing: normalisation, tokenisation, truecasing using Moses scripts, and BPE segmentation (Sennrich et al., 2016) with 36000 subwords. We tried training a model with 32 heads but could not due to memory constraints. For that reason, we start with a typical transformer-big (Vaswani et al., 2017) architecture using recommended hyperparameters. It has 16 heads of size 64 ($64 \times 16 = 1024$). Then, we train the same model but with 8, 4 and 2 heads of the same size. The results are below in Table 1.

Model	Heads	wmt14	wmt15	wmt16	Avg.
Transformer-big	16	26.7	29.8	33.9	30.1
Transformer-big-8	8	27.2	29.7	34.2	30.4
Transformer-big-4	4	26.1	29.0	34.2	29.8
Transformer-big-2	2	26.0	29.0	33.6	29.5

Table 1: A transformer-big with different number of heads for English→German.

When it comes to quality, the model needs a reasonable number of attention heads to perform well. The more this number is reduced, the worse the quality. However, more heads does not necessarily equal better translation quality. We concur that 8 heads per layer strikes a perfect balance between memory consumption and quality degradation.

5.2 Voita et al. (2019) pruning

Using the same language pair and dataset, we tried a pruning method presented by Voita et al. (2019). We used their Tensorflow implementation[2] with their training scripts, in which they set up a transformer-base architecture that it to be pruned globally. The pruning scheme requires a baseline model to fully converge first and then tuned with a regulariser that masks the heads. The attention sparsity is controlled by a λ hyperparameter.

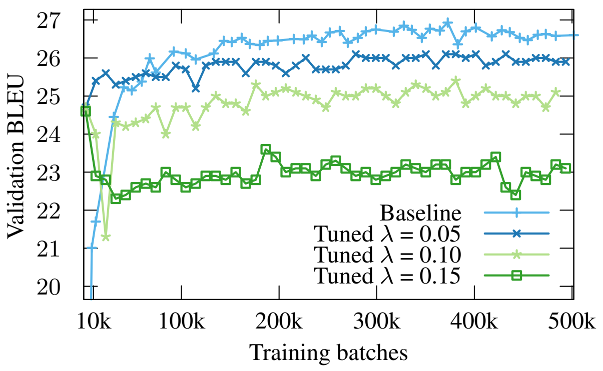

Figure 2: Validation BLEU for English→German transformer-base baseline and pruned with Voita et al. (2019) models.

The main focus of Voita et al. (2019) was attention analysis and its behaviour, rather than pruning and efficiency. Even though we used the authors'

[2] `https://github.com/lena-voita/the-story-of-heads`

Model	Sparsity	wmt14	wmt15	wmt16	Avg.	Δ
Baseline	0%	26.7	29.8	34.5	30.3	-
$\lambda = 0.05$	22%	26.4	29.7	34.0	30.0	-0.30
$\lambda = 0.10$	53%	25.1	27.9	31.8	28.3	-2.0
$\lambda = 0.15$	67%	23.5	25.8	28.8	26.0	-4.30

Table 2: Evaluation BLEU of English→German transformer-base models pruned with Voita et al. (2019)

implementation and the baseline achieved a reasonable score, pruning degraded its quality. Looking at Figure 2, the more sparsity was enforced with regularisation, the lower the translation quality. Even though we tuned for as long as the baseline training, the models do not recover. We tried experimenting with various hyperparameters settings such as learning rate and its scheduling, but to no further success.

5.3 Michel et al. (2019) pruning

Michel et al. (2019) experiment with pruning during and after training using a different heuristic: they introduce a mask variable for each head then define importance as the gradient of loss with respect to the mask variable. Their results are quite poor: pruning 40% of the total heads results in "staying within 85–90% of the original BLEU score". Results of pruning after training are worse: about 3 BLEU points lost with 40% sparsity and 10 BLEU points lost with 60% sparsity.[3] In our experiments, we see no loss in average BLEU at 67% sparsity.

We attribute our superior performance to adopting best practices for pruning during training (Frankle et al., 2019) and the choice of heuristic following Voita et al. (2019) instead.

Michel et al. (2019) reported that important heads emerge at the beginning of training. This supports our hypothesis that pruning during training will outperform pruning after training.

6 Setup

In order to investigate how effectively pruning works, we concentrate on two language pairs: Turkish→English and English→German. The first one is considered a low-resource, even with additional back-translated data. In contrast, English→German is a high-resource language pair with English not being a target language. We trained and decoded our models using the Marian machine translation toolkit (Junczys-Dowmunt et al., 2018a).

Turkish→English We use all the parallel data allowed by the constrained condition of the WMT18 (Bojar et al., 2018). The corpus consists of ~200 000 parallel sentences plus an additional 800 000 sampled from News Crawl and back-translated using a shallow NMT model trained on the existing small bilingual corpora (Haddow et al., 2018). We use the development and test sets provided in 2016. We also evaluate on the 2017 and 2018 testsets.

The preprocessing follows the steps of normalisation, tokenisation, truecasing using Moses scripts, and BPE segmentation (Sennrich et al., 2016). The vocabulary is shared and contains 36000 words. The architecture is transformer-big (Vaswani et al., 2017), trained using default recommended settings for such a model in Marian toolkit.[4] The models trained until cross-entropy has stopped improving for 10 consecutive validations, and select model checkpoints with highest BLEU scores.

English→German To measure impact on the speed of a highly optimized system, we follow the Workshop on Neural Generation and Translation 2020 Efficiency Shared task.[5] The shared task specified English→German translation under the WMT 2019 data condition (Barrault et al., 2019). As is standard for efficient translation, we applied teacher-student training (Kim and Rush, 2016) using the sentence-level system submitted by Microsoft to the WMT 2019 News Translation Task (Junczys-Dowmunt, 2019). The student models have a standard 6-layers transformer encoder (Vaswani et al., 2017) but the decoder is a faster two-layer Simpler Simple Recurrent Unit (SSRU) (Kim et al., 2019). The embedding dimension is 256, feed-forward network size is 1536. The models use shared vocabulary of 32,000 subword units created with SentencePiece (Kudo and Richardson, 2018).

All student models were trained on 13M sentences of available parallel data, using the concatenated English-German WMT testsets from 2016-2018 as a validation set.[6] The models were trained

[3]Comparisons are based on their reported numbers, which use non-standard tokenized BLEU (Post, 2018).

[4]Available via `--task transformer-big`.

[5]`https://sites.google.com/view/wngt20`

[6]The validation sentences were not teacher-translated.

until BLEU stopped improving for 20 consecutive validations to overfit the teacher, and the checkpoint with highest BLEU scores was selected. Since a student model should mimic the teacher as closely as possible, we did not use regularization like dropout and label smoothing. Other training hyperparameters were Marian defaults for training a Transformer Base model.[7] Student models have sharp probability distributions so we translate using beam size 1. Thanks to those settings, the baseline translates about 2335 words per second on a single CPU core.

7 Experiments

The goal is to prune as many heads as possible without damaging translation quality. The pruning procedure has some hyperparameters: the late resetting point, how long to train before making a pruning decision and how many heads to prune each iteration. Exploring this space is expensive; we arbitrarily set these to 5–6 saving checkpoints (25k batches for en–de, 12k for tr–en) Each pruning iteration have run for 3-4 checkpoints (15k batches for en–de, 8k for tr–en) after which selected attention heads are removed. The number of heads removed is roughly a total number of layers containing attention divided by 2. Removing less than that makes pruning slow and removing more in one go results in a unified distribution of attention heads (it usually picks one head per layer) and may be too aggressive in some cases. In each iteration, we change a seed value to make a model see data in different order.

We focus on results roughly within 50% to 85% heads removed. This range covers the interesting part from minor to noticeable degradation in translation quality. To evaluate an iteration, heads are pruned as usual then we reset the model back to the late resetting checkpoint and continue training to completion.

7.1 Transformer-big (Turkish→English)

Since we have shown that there is no need for having 16 heads per layer in transformer-big architecture (Section 5.1), we halve our attention matrices to start pruning from 8 heads per layer to save time. Thus, the model has 144 attention heads in total: 48 (6 layers with 8 heads each) self-attention heads in the encoder, 48 self-attention heads in the decoder, and 48 context heads in the decoder that attend to the encoder. The model was pretrained for 12k batches. Then, we train in a loop for 8k updates, remove 8 heads, revert and repeat until satisfied. The convergence progression is presented in Figure 3.

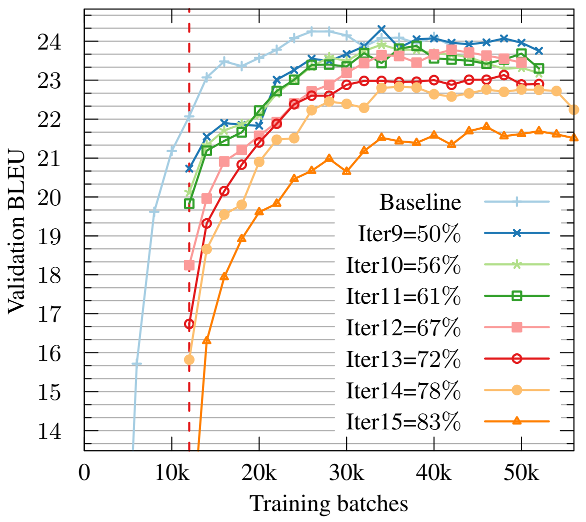

Figure 3: Convergence of Turkish→English models after removing a given percentage of all attention heads.

The baseline reaches the top BLEU scores quicker, but many pruned models still achieve competitive results later in training. The dashed vertical line shows the late resetting checkpoint. Pruning up to 61–67% (Iter. 11–12 in Figure 3) of the heads leads to longer convergence times, but nearly the same BLEU results on the development set. There is a breaking point of considerable damage at about 83% heads removed.

In Table 3, we perform evaluation and calculate the average difference in BLEU between the unpruned and pruned models. Similarily to training validation, pruning up to 72% of heads mostly maintains quality, then degrades progressively beyond that point.

7.2 Tiny student (English→German)

In this model, the decoder is already reduced to two tied layers. Since in self-attention is replaced with an SSRU anyway and context is not prioritised by our algorithm, we focus on pruning only the encoder.

We pretrained the model for 25k batches, with each pruning iteration lasting 15k updates and removing 3 heads from the encoder. The results are presented in Table 4. The models follow the trend set by our Turkish→English experiments — 75% of encoder heads can be removed with slight (-0.2)

[7]Available via `--task transformer-base`.

Model	Sparsity	Encoder	Context	Decoder	wmt16	wmt17	wmt18	Avg.	Δ
Baseline	0%	8 8 8 8 8 8	8 8 8 8 8 8	8 8 8 8 8 8	22.7	21.9	23.1	22.6	-
Pruned i9	50%	7 2 4 6 7 8	0 1 2 3 6 7	8 3 2 4 1 1	22.7	22.4	23.5	22.9	0.3
Pruned i10	56%	7 1 3 6 7 8	0 1 2 3 5 6	8 2 2 3 0 0	23.1	22.2	23.5	22.9	0.3
Pruned i11	61%	6 1 3 5 7 8	0 1 2 2 4 5	8 1 1 2 0 0	22.9	22.0	23.4	22.8	0.2
Pruned i12	67%	5 1 3 5 6 7	0 1 2 2 3 4	7 1 0 1 0 0	22.5	22.1	23.5	22.7	0.1
Pruned i13	72%	4 1 2 5 5 6	0 1 2 2 2 3	6 1 0 0 0 0	22.8	21.6	23.1	22.5	-0.1
Pruned i14	78%	3 1 2 5 4 5	0 0 2 2 1 2	5 0 0 0 0 0	22.0	21.3	22.6	22.0	-0.6
Pruned i15	83%	2 1 1 4 4 4	0 0 1 2 0 1	4 0 0 0 0 0	21.9	21.2	22.6	21.9	-0.7

Table 3: Evaluation of Turkish→English transformer-big models converged at i$^{\text{th}}$ pruning iteration with their distribution of attention heads.

Model	Params	Sparsity	Enc. heads	wmt16	wmt17	wmt18	wmt19	Avg.	Δ
Baseline	15,7M	0%	8 8 8 8 8 8	36.4	29.3	43.2	39.9	37.2	-
Pruned i9	14,8M	56%	5 1 3 1 5 6	36.5	29.1	43.4	40.0	37.3	0.1
Pruned i10	14,7M	63%	4 1 2 1 4 6	36.5	29.0	43.5	39.7	37.2	0.0
Pruned i11	14,6M	69%	3 0 2 1 4 5	36.3	29.0	43.3	39.9	37.1	-0.1
Pruned i12	14,5M	75%	2 0 2 1 3 4	36.3	28.8	43.1	39.8	37.0	-0.2
Pruned i13	14,4M	81%	1 0 2 1 2 3	36.3	28.9	42.7	39.5	36.9	-0.3
Pruned i14	14,3M	88%	0 0 1 1 1 3	35.7	28.4	42.1	38.9	36.3	-0.9

Table 4: Evaluation of English→German transformer student models converged at i$^{\text{th}}$ pruning iteration, with their distribution of attention heads.

damage to the quality. Pruning more than that is a trade-off between sparsity and quality.

In conclusion, the lottery ticket approach successfully pruned attention heads in both large transformer model and a tiny student architecture based on a simple heuristic; we leave the general case of block-sparse pruning to future work.

8 Analysis

In this section, we further analyse our pruning results in terms of pruning progress and head distribution. We reinitialise our pruned English→German models to demonstrate that the advantage of pruning comes from lucky initialisation, not the architecture itself.

Head distribution In both Table 3 and 4, we present attention distribution as it changes throughout pruning iterations. Each attention prioritised heads differently depending on layer depth and which attention type it is. Looking at Turkish→English results, the decoder attention is pruned more eagerly with more and more heads removed in each layer. The first layer seems to be crucial, others almost not at all. This seems to explain the trend of student models having 1–2 decoder layers and still performing well. The context attention interlocks with the decoder self-attention with each consecutive layer gaining more importance than the previous one. When it comes to the encoder in both language pairs, the middle layers do not hold the same significance as the first and last ones.

Architecture or initialisation? To check if the lottery ticket hypothesis is right in the context of our paper, we reinitialise our pruned models while keeping their structure. We compare average BLEU difference between pruned (Table 4) and trained from scratch (Table 5) models.

There is a consistent quality gap between pruned and reinitialised models that widens with sparsity. It confirms the assumptions made by the lottery ticket hypothesis: starting with a larger model and then deliberately selecting attention heads reveals which are the "winning tickets" in the initialisation lottery.

9 Speed

The main objective of our research is to remove heads from a transformer to make inference faster. For this reason, we make a trade-off between a total training time and inference speed, which is particularly useful in an industry production environment. In Table 6, we compare how long it takes to prune and train a model in comparison to the baseline approach. In practice, if a model trains for 2–3 days, an additional day is needed for a pruning procedure.

To compare translation speed, we select the

Model	Params	Sparsity	Enc. heads	wmt16	wmt17	wmt18	wmt19	Avg.	Δ
Baseline	15,7M	0%	8 8 8 8 8 8	36.4	29.3	43.2	39.9	37.2	-
Reinit i9	14,8M	56.25%	5 1 3 1 5 6	36.5	28.9	43.1	40.0	37.1	-0.1
Reinit i10	14,7M	62.50%	4 1 2 1 4 6	36.4	28.9	43.0	39.6	37.0	-0.2
Reinit i11	14,6M	68.75%	3 0 2 1 4 5	36.1	28.9	42.7	39.4	36.8	-0.4
Reinit i12	14,5M	75.00%	2 0 2 1 3 4	36.2	28.5	42.4	39.4	36.6	-0.6
Reinit i13	14,4M	81.25%	1 0 2 1 2 3	35.9	28.5	42.3	39.5	36.6	-0.6
Reinit i14	14,3M	87.50%	0 0 1 1 1 3	35.5	28.2	41.7	38.9	36.1	-1.1

Table 5: Evaluation of English→German student models that have the same pruned architecture as in Table 4 but with reinitialised parameters and trained from scratch. Lottery ticket pruning ensures better quality due to careful parameter selection which is nullified when reinitialised.

Model	Pretrain	Pruning	Convergence	Total
Baseline	-	-	475k	475k
Pruned 81%	25k	15k × 13	400k	620k

Table 6: The number of training updates in the baseline and the pruned English→German student model.

models with the best Pareto trade-off between quality and sparsity. The speed comparison is presented in Table 7.

Despite attention heads being just a small fraction of all parameters (~5% fewer parameters with about 10% size reduction), pruning them lessens the burden on inference significantly. Since all three attention types were pruned in transformer-big experiments, the speed-up is considerable — the model is 1.5 times faster with 0.3 BLEU loss.

In their paper among many reported models, Junczys-Dowmunt et al. (2018b) achieved 8.57× speed-up with −0.8 BLEU loss on GPU when scaling down from transformer-big teacher to transformer-base student. In another experiment, they gained 1.31× speed-up with −0.6 BLEU when using int8 quantisation on CPU. Our method is complementary to those as lottery ticket pruning can always remove heads on the top of existing solutions.

Continuing that line of thought, our small student model translates about 10% faster when pruned. However, it is important to remember that decoder is the key reason why the transformer is slow and it has already been optimized with an SSRU. This means there is a smaller margin of improvement in this type of a model. Again, attention pruning in this case is complementary and pushes the state-of-the-art even further. Just for comparison, we also include the baseline models trained with half (4) and one (1) attention head in each layer (including context attention). The model with just one head everywhere is slightly

(a) Turkish→English

transformer-big, wmt17, 1 GPU, beam 6, batch 32

Model	Sparsity	Params	Size	BLEU	Time
Baseline	0%	156.6M	670MB	21.9	46.75s
Pruned i13	72%	148.1M	566MB	21.6	31.55s

(b) English→German

transformer student, wmt19, 1 CPU, beam 1, batch 32

Model	Global sparsity	Params	Size	BLEU	Time
Baseline	0%	15.7M	61MB	39.9	18.11s
Baseline.half	50%	14.8M	57MB	39.0	17.98s
Baseline.one	88%	14.1M	54MB	37.8	15.15s
Pruned i12	64%	14.5M	56MB	39.8	16.24s

Table 7: Translation speed comparison between baseline and the best pruned models (converged at 13th and 12th pruning iterations in the respective models).

faster than our pruned model but at the cost of 2 BLEU points. This clearly shows again that careful pruning gives much better results than just training a smaller model from the start.

To compare our work with the state-of-the-art in machine translation speed, we submitted English→German student models to the WNGT2020 efficiency shared task (Bogoychev et al., 2020). These submissions were converged on a larger amount of data to maximize quality. Since our method usually selects one head to remove per layer, we experimented with more aggressive and lenient pruning by removing 3 and 6 heads per iteration respectively. These submissions were on the Pareto frontier for speed and quality, meaning that no other submission was simultaneously faster and higher quality.

The speed-up is about 10% on CPU with 75% encoder heads removed (Tab. 8). In terms of on GPU, our best pruned model gains 15% speed-up w.r.t. words per second (WPS) losing 0.1 BLEU in comparison to an unpruned model (Tab. 9). These results show that even when tested on a

				BLEU		
Model	Enc. heads	Params.	Size	WMT19	WMT1*	WPS
Tiny	8 8 8 8 8 8	15.7M	61MB	41.5	32.9	2050
Tiny.Steady.i12	2 0 2 1 3 4	14.5M	56MB	41.1	32.4	2282
Tiny.Steady.i14	0 0 1 1 1 3	14.3M	55MB	40.8	32.1	2350
Tiny.Pushy.i6	2 2 2 2 2 2	14.5M	56MB	41.4	32.4	2298
Tiny.Pushy.i7	1 1 1 1 1 1	14.3M	55MB	40.2	31.5	2346

Table 8: Quality and inference speed of our WNGT2020 models with pruned attention on CPU. Words per second (WPS) is evaluated in *float32* with a single CPU core on the official WNGT2020 input of 1M sentences.

				BLEU		
Model	Enc. heads	Params.	Size	WMT19	WMT1*	WPS
Tiny	8 8 8 8 8 8	15.7M	61MB	41.5	32.9	8210
Tiny.Steady.i12	2 0 2 1 3 4	14.5M	56MB	41.4	32.4	9518
Tiny.Pushy.i6	2 2 2 2 2 2	14.5M	56MB	41.0	32.4	9508

Table 9: Quality and inference speed of our WNGT2020 models with pruned attention on GPU. Words per second (WPS) measured on an AWS *g4dn.xlarge* instance with one NVidia T4 GPU.

larger scale, the pruned models achieve comparable quality with faster translation.

10 Future work

In this paper, we applied block-wise pruning to the transformer and its attention mechanism in particular. The natural progress of this research would be to prune other parts of the network — with the lottery ticket approach or not — to see how far block pruning can go without too much impact on quality. Furthermore, the heuristic algorithm we chose that decides which heads are not to be removed can definitely be improved on and extended to other types of block-sparsity cases.

11 Conclusions

This paper investigated block-wise pruning of attention heads in the transformer by applying the lottery ticket hypothesis to the problem. We used an iterative approach with pruning done in early stages training. Our experiments on NMT have proved that it is possible to remove a significant percentage of all heads (50–72%) in a large transformer with no significant damage to translation quality. Since attention mechanism is expensive, especially during inference, reducing the number of heads in a model led to $1.5\times$ speed-up and more if one is willing to sacrifice quality for speed. In the teacher-student regime, the student model with a reduced decoder can be pruned of 75% encoder heads with 0.1–0.2 BLEU loss and 10–15% faster translation speed. This shows that lottery ticket pruning is complementary to other methods that reduce inference workload. No matter how a model is trained like, attention heads can be easily removed from it.

We hope our paper will inspire further work on attention-sparse architectures. In our paper, we have only shown one example of a heuristic approach — there may be yet to be identified more efficient algorithms better suited to specific tasks, which will result in no need to train overly parametrised models.

Acknowledgements

This work was supported by the Engineering and Physical Sciences Research Council (grant EP/L01503X/1), EPSRC Centre for Doctoral Training in Pervasive Parallelism at the University of Edinburgh, School of Informatics.

This work has been performed using resources provided by the Cambridge Tier-2 system operated by the University of Cambridge Research Computing Service (www.hpc.cam.ac.uk) funded by EPSRC Tier-2 capital grant EP/P020259/1.

References

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019.

Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Nikolay Bogoychev, Roman Grundkiewicz, Alham Fikri Aji, Maximiliana Behnke, Kenneth Heafield, Sidharth Kashyap, Emmanouil-Ioannis Farsarakis, and Mateusz Chudyk. 2020. Edinburgh's submissions to the 2020 machine translation efficiency task. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*, pages 218–224, Online. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (WMT18). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels. Association for Computational Linguistics.

Christopher Brix, Parnia Bahar, and Hermann Ney. 2020. Successfully applying the stabilized lottery ticket hypothesis to the transformer architecture. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3909–3915, Online. Association for Computational Linguistics.

Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. 2019. Stabilizing the lottery ticket hypothesis. *CoRR*, abs/1903.01611.

Trevor Gale, Erich Elsen, and Sara Hooker. 2019. The state of sparsity in deep neural networks. *CoRR*, abs/1902.09574.

Maximilian Golub, Guy Lemieux, and Mieszko Lis. 2018. Dropback: Continuous pruning during training. *CoRR*, abs/1806.06949.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2019. Minimally-augmented grammatical error correction. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 357–363, Hong Kong, China. Association for Computational Linguistics.

Barry Haddow, Nikolay Bogoychev, Denis Emelin, Ulrich Germann, Roman Grundkiewicz, Kenneth Heafield, Antonio Valerio Miceli Barone, and Rico Sennrich. 2018. The university of Edinburgh's submissions to the WMT18 news translation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 399–409, Belgium, Brussels. Association for Computational Linguistics.

Marcin Junczys-Dowmunt. 2018. Microsoft's submission to the WMT2018 news translation task: How I learned to stop worrying and love the data. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 425–430, Belgium, Brussels. Association for Computational Linguistics.

Marcin Junczys-Dowmunt. 2019. Microsoft translator at WMT 2019: Towards large-scale document-level neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 225–233, Florence, Italy. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018a. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Kenneth Heafield, Hieu Hoang, Roman Grundkiewicz, and Anthony Aue. 2018b. Marian: Cost-effective high-quality neural machine translation in C++. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 129–135, Melbourne, Australia. Association for Computational Linguistics.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Young Jin Kim, Marcin Junczys-Dowmunt, Hany Hassan, Alham Fikri Aji, Kenneth Heafield, Roman Grundkiewicz, and Nikolay Bogoychev. 2019. From research to production and back: Ludicrously fast neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 280–288, Hong Kong. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 14014–14024. Curran Associates, Inc.

Sharan Narang, Eric Undersander, and Gregory F. Diamos. 2017. Block-sparse recurrent neural networks. *CoRR*, abs/1711.02782.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Abigail See, Minh-Thang Luong, and Christopher D. Manning. 2016. Compression of neural machine translation models via pruning. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 291–301, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Tong Xiao, Yinqiao Li, Jingbo Zhu, Zhengtao Yu, and Tongran Liu. 2019. Sharing attention weights for fast transformer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5292–5298. International Joint Conferences on Artificial Intelligence Organization.

Haonan Yu, Sergey Edunov, Yuandong Tian, and Ari S. Morcos. 2020. Playing the lottery with rewards and multiple languages: lottery tickets in rl and nlp. In *International Conference on Learning Representations*.

Michael Zhu and Suyog Gupta. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression.