Aligned Weight Regularizers for Pruning Pretrained Neural Networks

Anonymous ACL submission

Abstract

While various avenues of research have been explored for iterative pruning, little is known what effect pruning has on zero-shot test performance and its potential implications on the choice of pruning criteria. This pruning setup is particularly important for cross-lingual models that implicitly learn alignment between language representations during pretraining, which if distorted via pruning, not only leads to poorer performance on language data used for retraining but also on zero-shot languages that are evaluated. In this work, we show that there is a clear performance discrepancy in magnitude-based pruning when comparing standard supervised learning to the zero-shot setting. From this finding, we propose two weight regularizers that aim to maximize the alignment between units of pruned and unpruned networks to mitigate alignment distortion in pruned cross-lingual models for that performs well for both non zero-shot and zero-shot settings. We provide experimental results on cross-lingual tasks for the zero-shot setting using XLM-RoBERTa$_{\text{Base}}$, where we also find that pruning has varying degrees of representational degradation depending on the language corresponding to the zero-shot test set. This is also the first study that focuses on cross-lingual language model compression.

1 Introduction

Deep neural networks (DNNs) have grown increasingly large in the recent years. This has led to models requiring more storage requirements, more resources for training and inference (e.g., GPUs and TPUs), longer compute times and larger carbon footprints. This is largely due to the rise of masked self-supervised learning (SSL) which trains DNNs (e.g., Transformers in NLP) on a large collection of samples that do not have task labels but instead use a subset of the inputs as labels. Given the aforementioned challenges, it has become more difficult for machine learning practitioners to use these SSL pretrained models for fine-tuning on downstream tasks. While training tricks such as effective batch sizes, gradient accumulation and dynamic learning rate schedules (Howard and Ruder, 2018) have improved the efficiency of fine-tuning DNNs under resource constraints, it can still come at a cost, e.g. gradient accumulation leads to less updates.

Pruning (LeCun et al., 1990; Reed, 1993) is a type of model compression method (Buciluǎ et al., 2006) that aims to address these shortcomings by zeroing out a subset of weights in the DNN, while maintaining performance close to the original model. Retraining is often carried out directly after each pruning step to recover from pruning induced performance drops. This process is referred to as *iterative pruning*. Although, iterative pruning has been extensively studied in the SSL setting (Hassibi and Stork, 1993; Han et al., 2015a; Ding et al., 2018) and the transfer learning setting (Molchanov et al., 2016; Gordon et al., 2020; Sanh et al., 2020), little is known about pruning DNNs in the zero-shot setting[1] where a model is required to make predictions on a set of samples from classes that are unobserved during training. One salient example is pretrained cross-lingual language models (XLMs) (Lample and Conneau, 2019; Conneau et al., 2019) whereby the model is trained with a masked/translation language model (MLM/TLM) objective to predict tokens for a large set of different languages whereby the objective forces the XLM model to learn similar representations for different languages. After cross-lingual pretraining, the model is further fine-tuned to a downstream task in one language (e.g., English) and then evaluated on different languages in the zero-shot setting (e.g., Spanish, French, Chinese, etc.). In this context, applying current pruning methods can damage the

[1] Here, zero or one-shot is the conventional usage of the meaning (i.e., number of samples per class), not one-shot pruning (2018) which is the number of pruning steps used during retraining.

XLM cross-lingual alignment that has been learned during pretraining. Ideally, we would aim to prune XLMs in such a way that avoids this alignment distortion which effects the zero-shot performance of pruned XLMs. Additionally, overfitting to the language used for fine-tuning becomes more of an issue due to the progressive reduction in parameters throughout iterative pruning as the remaining weights are relatively large, moving away from an "aligned" XLM state.

This is an important problem to address as the application of large pretrained models in the zero shot-setting for both natural language and computer vision is of practical importance e.g., using XLMs in production for multiple languages by only requiring annotations in a single language for fine-tuning, making predictions on unseen classes at test time from pretrained visual representations (Bucher et al., 2017) using only semantic descriptions (i.e., label similarity to known classes) or zero-shot predictions in pretrained multi-models such as CLIP (Radford et al., 2021).

Hence, this work addresses the *alignment distortion* pruning problem by introducing *AlignReg*, a class of weight regularizers for magnitude-based pruning that force pruned models to have parameters that point in a similar direction or have a similar distribution to the parameters of the original pretrained network. To our knowledge, this is the first study on how iteratively pruned models perform in the zero-shot setting and how the solution differs from solutions found in the non-zero shot setting. We believe our findings have a strong practical implication as well-established pruning criteria may not be suitable given the observed discrepancy between zero-shot performance and the typically reported non-zero shot performance. Moreover, our proposed weight regularizer improves overall pruning generalization in zero-shot cross-lingual transfer. Below, we summarize our **contributions**.

- The first analysis of pruning cross-lingual models, how this effects zero-shot cross-lingual transfer and performance differences to pruning in the SSL setup.
- A weight regularizer that mitigates alignment distortion by minimizing the *layer-wise Frobenius norm or unit similarity* between the pruned model and unpruned model, avoiding overfitting to single language task fine-tuning.
- A post-analysis of weight distributions after pruning and how they differ across module types in Transformers.

2 Related Work

Below we describe regularization-based pruning, other non-magnitude based pruning and how masked language modeling (MLM) implicitly learns to align cross-lingual representations.

Regularization-based pruning. Pruning can be achieved by using a weight regularizer that encourages network sparsity. Three well-established regularizers are L_0 (Louizos et al., 2017), L_1 regularization (Liu et al., 2017; Ye et al., 2018) and the commonly used L_2 regularization for weight sparsity (Han et al., 2015b,a). Wang et al. have proposed an L_2 regularizer that increases in influence throughout retraining and shows the increasing regularization improves pruning performance. For structured pruning where whole blocks of weights are removed, Group-wise Brain Damage (Lebedev and Lempitsky, 2016) and SSL (Wen et al., 2016) propose to use Group LASSO (Yuan and Lin, 2006) to learn structured solutions.

Importance-based pruning. Magnitude-based pruning (MBP) relies on the assumption that weight or gradient magnitudes have correlation with its importance to the overall output of the network. Mozer and Smolensky instead use a learnable gating mechanism that approximates layer importance, finding that weight magnitudes reflect importance statistics. To measure weight importance as the difference in loss between pruned and unpruned network, LeCun et al. approximate this difference with a Taylor series up to the second order. This involves the product of the gradient and weight magnitude in the 1st term and an approximation of the Hessian and the square of the weight magnitude for the second term. However, computing the Hessian and even its approximations (LeCun et al., 1990; Hassibi and Stork, 1993; Dong et al., 2017; Wang et al., 2019; Singh and Alistarh, 2020) can significantly slow down retraining. In our work, we avoid the requirement of computing the Hessian or approximations thereof, as it is not scalable for models such as XLM-R (Conneau et al., 2019). Park et al. have extended MBP to block approximations to avoid pruning lowest weight magnitudes that may be connected to weights in adjacent layers that have high weight magnitude. Lee et al. have provided a method to automatically choose the sparsity of layers by using the rescaled version

of weight magnitude to incorporates the model-level distortion incurred by pruning.

Implicit Alignment in Pretrained MLMs In context of multi-task learning, Chen et al. (2020) minimize the mean squared error between pretrained weights and weights being learned for a set of different source tasks to avoid catastrophic forgetting in the continual learning setting. Wu et al. (2019) have found that multilingual MLM (i.e training with an MLM objective with concatenated text for multiple languages) naturally leads to models with strong cross-lingual transfer capabilities. Additionally, they find that this is also found for monolingual models that do not share vocabulary across monolingual corpora and the only requirement is that weight sharing is used in the top layers of the multi-lingual encoder. In the context of our work, we want to bias our fine-tuned and iteratively pruned model to have similar geometric properties and symmetries to these pretrained MLMs to preserve zero-shot cross-lingual transfer.

3 Methodology

In this section, we describe how our proposed *AlignReg* weight regularizers can improve pruning performance in both supervised learning and zero-shot pruning settings. We focus on two regularizers, namely, *a neuron correlation-based regularizer* (cosine-MBP) and *Frobenius layer-norm regularizer* (frobenius-MBP).

Let $\mathcal{D} := \{X_i, y_i\}_{i=1}^{D}$ where each X_i of D training samples consists of a sequence of vectors $X_i := (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ and $\boldsymbol{x}_i \in \mathbb{R}^d$ (e.g., $d = 512$). For structured prediction (e.g., NER and POS), $y_i \in \mathbb{R}^{n \times c}$ and for single and pairwise sentence classification, $y_i \in \mathbb{R}^c$ where c is the number of classes. Let $\theta = (\theta_1, \ldots, \theta_L)$ be the parameters of a pretrained network f with L layers, where θ_l refers to the parameters, including weight matrix $\mathbf{W}_l$ and bias b_l, at layer l. Let $f_{\tilde{\theta}}$ be a network with parameters $\tilde{\theta}$ consisting of weights $\tilde{\mathbf{W}}_l \in \mathbb{R}^{N_{l-1} \times N_l}$ and bias $\tilde{\boldsymbol{b}}_l \in \mathbb{R}^{N_l}$ where N_l is the number of units in the l-th layer. Here, $\tilde{\mathbf{W}}_l := \mathbf{W}_l \mathbf{M}_l$ where $\mathbf{M}$ is the pruned mask. For MBP (Karnin, 1990) we remove weights of $\mathbf{W}_l$, $\forall l \in L$ with the smallest absolute weight magnitude until a specified percentage p of weights are removed. Note that this is a layer-wise process and requires the pruned weights to be masked with $\mathbf{M}_l$ which has 0 entries corresponding to weights to be pruned and 1 entries for unpruned weights $\mathbf{W}_l$. Global MBP can also be used whereby the weights $\{\mathbf{W}_l\}_{l=1}^{L}$ are first vectorized and concatenated prior to choosing p lowest weight magnitudes. Unlike layer-wise MBP, the percentage of weights removed in each layer can vary for global-MBP. Typically, weight regularization is used with MBP to encourage weight sparsity. Thus the objective for iterative pruning pruning can be expressed as,

$$\mathcal{L}_\theta := \mathbb{E}_{\boldsymbol{z}}\Big[\frac{1}{D}\sum_{i=1}^{D} \ell_{ce}\big(f_{\tilde{\theta}}(\mathbf{X}_i), \boldsymbol{y}_i\big) + \lambda||\tilde{\theta}||_0\Big] \quad (1)$$

where λ controls the influence of the weight magnitude regularization. We now describe our proposed *AlignReg*.

3.1 *AlignReg* - Pruning-Aware Regularization

AlignReg can be used to align weights unit-wise or layer-wise between unpruned and pruned networks. We initially discuss the cosine-MBP regularizer.

cosine-MBP aims to preserve the inherent cross-lingual alignment, during iterative pruning, by minimizing the angle between parameter vectors of the same unit in the pruned and unpruned network. The intuition is that cross-lingual alignment relies more on parameter vector direction than vector magnitudes. Moreover, as the network is being pruned, the weights will consequently change weight magnitude to account for the information loss. To apply *AlignReg* to linear layers within Transformers, we compute the pairwise cosine similarity between pairs of pruned weights $\tilde{\mathbf{W}}_l \subset \tilde{f}$ and unpruned weights $\mathbf{W} \subset f$ for all l-th layers. For $\mathbf{W}_l \in \mathbb{R}^{N_{l-1} \times N_l}$ of the l-th layer, the average weight correlation is

$$\rho(\tilde{\mathbf{W}}_l, \mathbf{W}_l) = \frac{1}{N_l}\sum_{i=1}^{N_l} \frac{|\mathbf{W}_{li}^\top \tilde{\mathbf{W}}_{li}|}{||\mathbf{W}_{li}||_2 ||\tilde{\mathbf{W}}_{li}||_2} \quad (2)$$

where $\mathbf{W}_{li}$ is i-th column of the matrix corresponding to the i-th unit of the l-th layer. Intuitively, $\rho(\mathbf{W}_l, \tilde{\mathbf{W}}_l)$ is the average cosine similarity between weight vectors of the same unit at the l-th layer of the pruned and unpruned network. Adding *AlignReg* to the objective results in Equation (3),

$$\mathcal{L}_\theta := \ell_{ce}\big(f_{\tilde{\theta}}(\mathbf{X}), \boldsymbol{y}\big) + \frac{\lambda}{L}\sum_{l}^{L} \rho\big(\tilde{\mathbf{W}}_l, \mathbf{W}_l\big) \quad (3)$$

where $\lambda \in [0, \infty)$ controls the importance of AlignReg relative to the main cross-entropy loss $\ell_{ce}(\cdot, \cdot)$. The gradient of the loss w.r.t to θ is then

expressed as equation (4),

$$\nabla_\theta \mathcal{L}_\theta := \nabla_{\tilde{\theta}} \ell_{ce}(f_{\tilde{\theta}}(\mathbf{X}), \boldsymbol{y}) + \frac{\lambda}{L} \sum_{l}^{L} \frac{\partial \rho(\tilde{\mathbf{W}}_l, \mathbf{W}_l)}{\partial \tilde{\mathbf{W}}_l} \quad (4)$$

where $\frac{\partial \rho(\tilde{\mathbf{W}}_l, \mathbf{W}_l)}{\partial \tilde{\mathbf{W}}_l}$ is a function of the '2-norm of the matrices in $\mathbf{W}_l$. For the element $\mathbf{W}_{l,(i,j)}$ of i-th row and j-th column in $\mathbf{W}_l$, we have

$$\frac{\partial \rho(\tilde{\mathbf{W}}_l, \mathbf{W}_l)}{\partial \tilde{\mathbf{W}}_{l,(i,j)}} = \frac{1}{N_l - 1} \sum_{j=1}^{N_l} \Big(\text{sign}(\mathbf{W}_{l,(j)}^\top \tilde{\mathbf{W}}_{l,(j))}) \Big[\frac{\tilde{\mathbf{W}}_{l,(i,j)}}{||\mathbf{W}_{l,(,j)}||_2 ||\tilde{\mathbf{W}}_{l,(,j)}||_2} - \frac{\mathbf{W}_{l,(i,j)} \mathbf{W}_{l,(,j)}^\top \tilde{\mathbf{W}}_{l,(,j)}}{||\mathbf{W}_{l,(,j)}||_2^3 ||\tilde{\mathbf{W}}_{l,(,j)}||_2} \Big] \Big) \quad (5)$$

where $\mathbf{W}_{l,(,j)}$ and $\tilde{\mathbf{W}}_{l,(,j)}$ are j-th column in $\mathbf{W}_l$ and $\tilde{\mathbf{W}}_l$, respectively. Thus, this regularization favors solutions with high cosine similarity between units of pruned and unpruned networks. We also consider a layer-wise $\rho(\mathbf{W}, \tilde{\mathbf{W}})$ that relaxes the unit-level alignment to whole layers. This is partially motivated due to the fact neural networks can exhibit similar output activation behavior even when neuron weights have been permuted within the layer (Brea et al., 2019). To perform this we simply apply Equation (2) with vectorized weights $\rho(\text{vec}(\tilde{\mathbf{W}}_l), \text{vec}(\mathbf{W}_l))$ and the subsequent partial derivatives in Equations (4) and (5) are applied for updating $\tilde{\mathbf{W}}_l$. In our experiments we did not see a significant difference using vectorized weights and thus use unit-wise cosine similarity.

Relaxing Unit-Wise AlignReg To A Layer-Wise Frobenius Distortion Formulation Thus far we have described the application of cosine similarity as a measure of similarity between unpruned and pruned weights of the same units. However, this may be a strict constraint, particularly at high compression rates where the remaining weights for a unit are forced to have higher norms to allow zeroed weights. Hence, an alternative measure is the layer-wise Frobenius norm (Frobenius-MBP) regularizer based on the difference between weights $||\mathbf{W} - \tilde{\mathbf{W}}||_F$. MBP itself can be viewed in terms of minimizing the Frobenius distortion (Han et al., 2015a; Dong et al., 2017) as $\min_{\mathbf{M}:||\mathbf{M}||_0=p} ||\mathbf{W} - \mathbf{M} \odot \mathbf{W}||_F$ where $\odot$ is the Hadamard product, $||\cdot||_0$ denotes the entrywise 0-norm, and p is a constraint of the number of weights to remove as a percentage of the total number of weights for that layer. In the zero-shot setting, we need to account for out-of-distribution Frobenius distortions, such as *alignment distortion* in XLM due to pruning and overfitting to a single language. Taking the view of Frobenius distortion minimization when using our weight regularizer, we reformulate it to include Frobenius-MBP as,

$$\min_{\mathbf{M}:||\mathbf{M}||_0=p} \Big[||\mathbf{W}\text{-}\mathbf{M} \odot \mathbf{W}||_F^2 + \lambda ||\mathbf{W}^T\text{-}\mathbf{M} \odot \mathbf{W}||_F^2 \Big] \quad (6)$$

where $\mathbf{W}^T$ are the weights from the pretrained model prior to fine-tuning that is cross-lingually aligned from the masked language modeling (MLM) pretraining objective. In our experiments, $\lambda = 5 \times 10^{-4}$.

***frobenius*-MBP Implicitly Aligns Eigenvectors** To explicitly show that the Frobenius distortion minimization aligns fine-pruned and pretrained parameter vectors we expect their eigenvectors to also be close. We can use the Eckart-Young-Mirsky Theorem (Golub et al., 1987) to express Frobenius distortion minimization as Equation 7,

$$||\mathbf{W}^T - \mathbf{M} \odot \mathbf{W}||_F^2 = ||\mathbf{\Sigma} - \mathbf{U}^\top \mathbf{M} \odot \mathbf{W}\mathbf{V}||_F^2 \quad (7)$$

where the unitary invariance under the 2-norm that $\mathbf{U}$,$\mathbf{V}$ vanishes and singular value matrix is left to approximate $\mathbf{W}^T$, hence the inclusion of $\mathbf{\Sigma}$. We express $\mathbf{X} = \mathbf{U}_k \Sigma_k^{12}$, $\mathbf{Y} = \Sigma_k^{12} \mathbf{V}_k^\top$ and $\mathbf{XY} = \mathbf{A}_k$. Hence, we can further describe the minimization as $||\Sigma - \mathbf{U}^\top \mathbf{W}_k^T \mathbf{V}||_F^2$ and since $\mathbf{X}$, $\mathbf{Y}$ are unitary, $||\Sigma - \Sigma_k||_F^2$.

3.2 Connections to Knowledge Distillation

Knowledge distillation (KD) works by using outputs of the last layer (Hinton et al., 2015) or intermediate layers (Romero et al., 2014) as additional soft targets. *AlignReg* regularizers instead operate directly on minimizing a divergence or distance between weight tensors as opposed to their corresponding output activations. Hence, *AlignReg* does not necessarily need training data as it operates directly on aligning weight tensors. Since the networks that are used for alignment are architecturally identical, we can show that maximizing weight similarity is equivalent to minimizing distance between their corresponding output activations (Romero et al., 2014) when the norm of input Z is smaller than the output range of σ. For our experiments, we use XLM-RoBERTa$_{\text{Base}}$ which contain Gaussian Linear Error Unit (GeLU) activation functions, which can be formulated as $\sigma(\mathbf{Z}_{li}) := \mathbf{Z}_{li}/2(1.0 + \text{erf}(\mathbf{Z}_{li}/\sqrt{2.0}))$ where erf is an error function, $\sigma(\cdot)$ is a monotonic activation function and $\boldsymbol{Z}_{li}$ is the input vector. The GELU

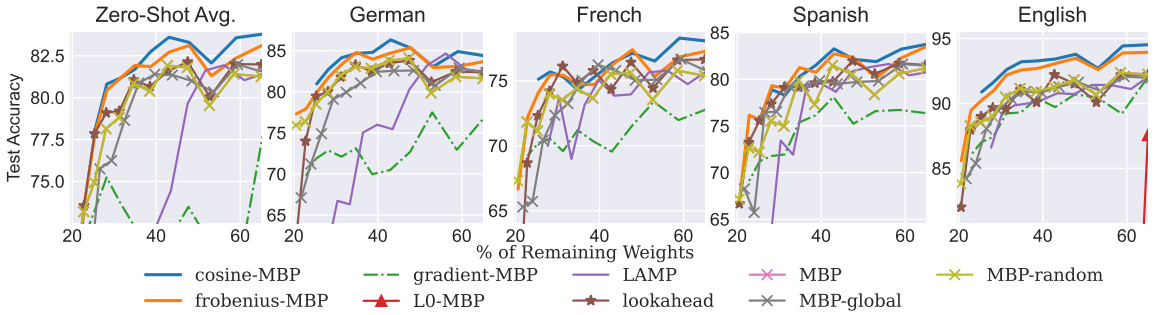

Figure 1: **English and Zero-Shot Test Accuracy on News Classification.**

activation has the properties that for $\mathbf{Z}_{li} > 0$ it is equivalent to the ReLU activation and $\mathbf{Z}_{li} \leq 0$ it tends to -1. For $\mathbf{Z}_{li} > 0$, $||\mathbf{Z}_{li}||_2 \leq 1$ and a monotonic piecewise linear function $\sigma(\cdot)$, the inequality $||\mathbf{W}_{li}\text{-}\mathbf{M}_{li} \odot \mathbf{W}_{li}||_F \leq ||\sigma(\mathbf{Z}_l\mathbf{W}_{li})\text{-}\sigma(\mathbf{Z}_{li}\mathbf{M}_{li} \odot \mathbf{W}_{li})||_F$ holds. Layer normalization leads to features having zero mean and unit variance and hence $||\mathbf{Z}_{li}||_2 \leq 1$. Hence, minimizing the Frobenius distortion of pruned and unpruned weights is equivalent to minimizing the mean squared error (MSE) between output activations, as is the knowledge distillation method used for FitNets (Romero et al., 2014). In contrast, KD using FitNets encourages the student network to have activation outputs that are the same as the teacher with permutation invariance on the units incoming weights, not restricting the weights to be similar. Unlike KD, this minimization can be peformance without any data.

4 Experimental Setup

Datasets. We perform experiments on multilingual tasks from the XGLUE benchmark (Liang et al., 2020) with pretrained $\text{XLMR}_{\text{Base}}$. This covers pairwise classification (QAM, QADSM, WPR, XNLI), sentence classification (NC) and structured prediction (NER and POS) tasks.

Iterative Pruning Details. Texts are tokenized using the SentencePiece BPE tokenizer (Sennrich et al., 2015) with a vocabulary of 250K tokens. For structured prediction tasks (POS and NER), a single layer feed-forward (SLFF) token-level classifier is used on top of $\text{XLM-R}_{\text{Base}}$ and for sentence-level task a SLFF sentence-level classifier is used. The batch size is 32, the learning rate is $5 \cdot 10^{-6}$ and the maximum sequence length is set to 256 for all tasks, except for POS in which we use a learning rate of $2 \cdot 10^{-5}$ and max sequence length of 128. We carry out a pruning step after each 15 training epochs, uniformly pruning 10% of the parameters at each pruning step. We omit the pruning of embedding layers, layer normalization parameters and the classification layer as they account for a relatively small number of the total parameter count ($< 1\%$) and play an important role in XLM generalization. Although prior work has suggested non-uniform pruning schedules (e.g., cubic schedule (Zhu and Gupta, 2017)), we did not see major differences to uniform pruning in preliminary experiments. Each task is trained with English data only and evaluated on all available languages for that task. Hence, we expect the percentage of achievable compression to be lower as performance in the zero-shot cross-lingual setting to be more difficult than the monolingual setting (e.g., GLUE tasks).

Pruning Baselines. Below lists our pruning baselines. **Random Pruning** (1997) - weights are pruned uniformly at random across all layers to a chosen fraction.**Layer-wise Magnitude Pruning** (MBP) (Janowsky, 1989; Mozer and Smolensky, 1989) - for each layer, weights with the lowest absolute value (LAV) are pruned. **Layer-wise Gradient Magnitude Pruning** (Sun et al., 2017) - for each layer, prunes the weights with LAV of the accumulated gradients evaluated on a batch of inputs. **Global Magnitude Pruning** (Global-MBP) (Karnin, 1990) - prunes weights with LAV anywhere in the DNN. L_0 **norm MBP** (Louizos et al., 2017) - uses non-negative stochastic gates that choose which weights are set to zero as a smooth approximation to the non-differentiable L_0-norm. **Lookahead pruning (LAP)** (Park et al., 2020) - prunes paths that have smallest weight magnitude across blocks of layers, unlike MBP which treats layers independently. **Layer-Adaptive Magnitude Pruning (LAMP)** (Lee et al., 2020) adaptively sets the pruning ratio of each layer.

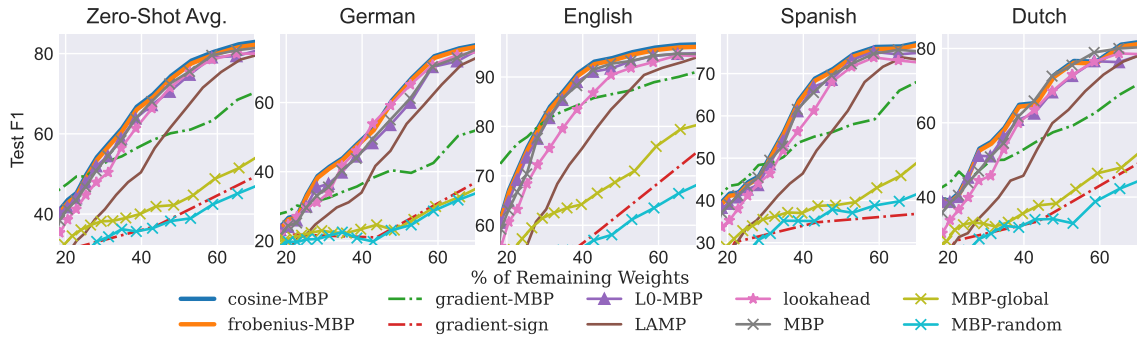

Figure 2: **Zero-Shot Test F1 on Named Entity Recognition.**

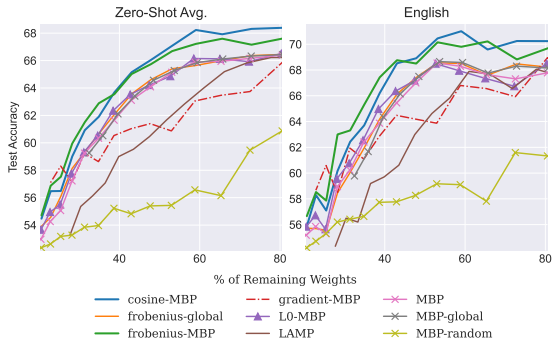

Figure 3: **Question Answer Matching Test Accuracy.**

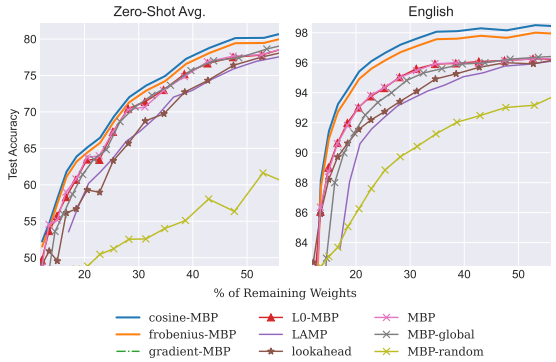

Figure 4: **Part of Speech Tagging Test Accuracy.**

5 Empirical Results

We now discuss results on the XGLUE tasks.

News Classification (NC) Figure 1 shows the results on news classification where a category for news article is predicted and evaluated in 5 languages and trained and iteratively pruned on English text. Firstly, we observe the trend in iterative pruning performance degradation is somewhat volatile. From preliminary experiments we found news classification to require only 3 epochs to converge for standard fine-tuning on XLM-RoBERTa$_{\text{Base}}$. We find that this task is relatively "similar" to the pretraining task and therefore able to easier recover from pruning steps. Overall, both Cosine-MBP and Frobenius-MBP consistently lead to the best zero-shot test performance across both pruning steps and languages.

Question Answer Matching (QAM) Figure 3 shows the test accuracy on English and the zero-shot test accuracy on French and German for Question-Answer Matching (QAM). This involves predicting whether a question is answered correctly or not given a question-answer pair. We find that Frobenius-MBP and Cosine-MBP maintain higher accuracy across multiple pruning steps, outperforming baselines. More generally, we see there is close to 2% drop in average test accuracy drop in French and German when compared to testing on samples from the same language used in training.

Named Entity Recognition (NER) The Named Entity Recognition (NER) cross-lingual dataset is made up of CoNLL-2002 NER and CoNLL-2003 NER (Sang and De Meulder, 2003), covering English, Dutch, German and Spanish with 4 named entities. From Figure 2 we find that cross-lingual transfer of pruned models is most difficult in German and Dutch, which both come from the same language family, sharing commonalities such as word order and having similar vocabularies. The primary reason for the difficulty in maintaining performance in high compression rates for this NER dataset is that there is only 15k training samples, being significantly lower than the remaining XGLUE tasks (the majority contains 100k training samples). Thus, not only is there less training data to recover directly after each pruning step, but the pruning step interval itself is shorter. In contrast, English test performance is close to the original performance up until 25% of remaining weights, unlike the remaining languages. We find that gradient-MBP eventually overtakes MBP approaches past

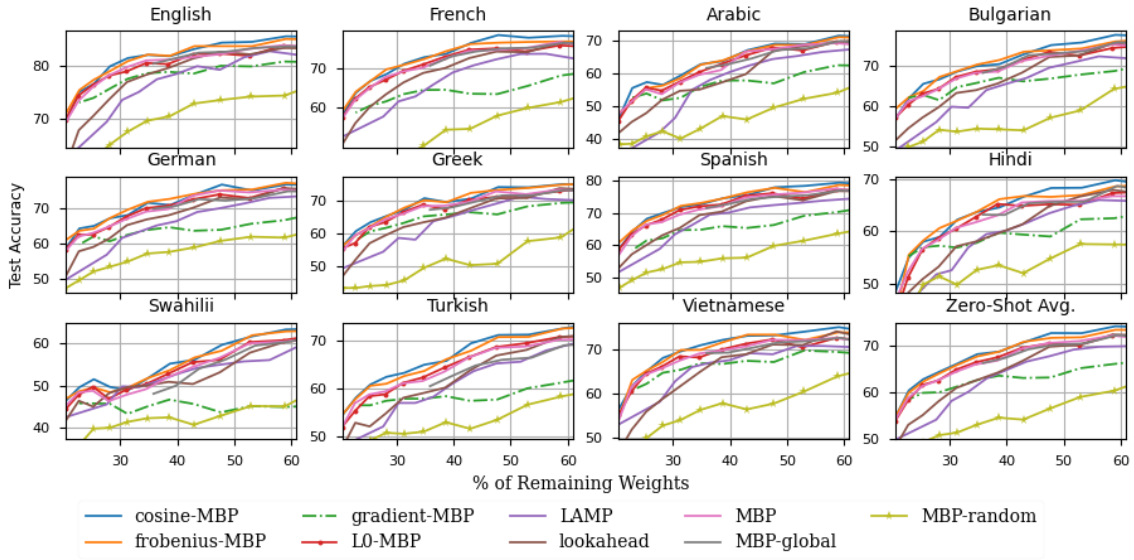

Figure 5: **Zero-Shot XNLI Results Per Language After Iteratively Fine-Pruning XLM-RoBERTa$_{\text{Base}}$**

20% remaining weights. However accuracy has reduced too much at this compression level. We find that Cosine-MBP and Frobenius-MBP weight regularizers achieve the best performing pruned model performance above 20% remaining weights, with Lookahead pruning and L_0 regularized MBP being competitive in zero-shot performance.

Part of Speech Tagging (POS) The Part of Speech (PoS) tagging dataset consists of a subset of the Universal Dependencies treebank (Nivre et al., 2020) and covers 18 languages. In Figure 4, we see both Cosine-MBP and Frobenius-MBP tend to outperform baselines, although L_0-based pruning (Louizos et al., 2017) has similar performance to Cosine-MBP for zero-shot accuracy. There is also a clear discrepancy between SSL accuracy (English) versus zero-shot accuracy (Average), the latter following closer to linear decay after 40-50% of weights remaining.

Cross-Lingual Natural Language Inference (XNLI) Figure 5 shows the zero-shot cross-lingual transfer for various unstructured pruning methods. We find that both the accuracy on the English test (i.e SSL generalization) and the average zero-shot test accuracy are consistently improved using Cosine-MBP and Frobenius-MBP, outperforming L_0 pruning, Lookahead pruning and LAMP. We find that morphologically rich languages such as Arabic, Swahili and Turkish degrade in performance linearly once performance begins to drop after 60% of the remaining weights are pruned. This trend is roughly followed for all MBP-based pruning methods. Additionally, test accuracy on English can be maintained within 10% accuracy drop of the original test accuracy up to 20% of remaining weights for MBP, while Swahili can only be within a 10% accuracy drop up to 55% of the remaining weights. Hence, iterative pruning in the zero-shot setting leads to faster performance degradation for languages that are typologically or etymologically further from the language used for fine-tuning.

When comparing, English and the average zero-shot test accuracy we see that the slope is steeper after the inflection point[2] for all pruning methods, not to mention the greater than 10% accuracy drop across pruning steps.

XGLUE Average Result Finally, in Table 1 we show the overall and average task *understanding* scores on the XGLUE benchmark for our proposed *AlignReg* weight regularizer and the pruning baselines. We find that the use of *AlignReg* Cosine-MBP and Frobenius-MBP better preserves cross-lingual alignment during model pruning, thereby outperform other MBP baselines, including LAMP and Lookahead pruning, based on improved zero-shot cross-lingual performance.

Discussion From our experiments, we found that layer-wise pruning tends to outperform global pruning. This can be explained by the clear discrepancy between weight norms of different layer types within each self-attention block. Global pruning

[2]The point which the performance slope significantly steepens and drops are relatively large to previous pruning steps.

Prune Method	XNLI	NC	NER	PAWSX	POS	QAM	QADSM	WPR	Avg.
No Pruning	73.48	80.10	82.60	89.24	80.34	68.56	68.06	73.32	76.96
Random	51.22	70.19	38.19	57.37	52.57	53.85	52.34	70.69	55.80
Global-Random	50.97	69.88	38.30	56.74	53.02	54.02	53.49	69.11	55.69
L_0-MBP	64.75	78.98	56.22	72.09	71.38	59.31	53.35	71.70	65.97
L_2-MBP	64.30	78.79	54.43	77.99	70.68	59.24	60.33	71.52	67.16
L_2-Global-MBP	64.17	78.64	54.47	75.51	72.27	59.26	60.10	71.50	66.99
L_2-Gradient-MBP	61.11	73.77	53.25	79.56	65.89	57.35	59.33	71.59	65.23
Lookahead	60.84	79.18	54.44	71.05	68.76	55.94	53.41	71.26	64.36
LAMP	58.04	63.64	51.92	66.05	67.43	55.36	52.42	71.09	60.74
Cosine-MBP	66.20	79.15	55.62	78.45	71.62	57.56	61.37	72.51	**67.81**
Frobenius-MBP	65.71	79.84	55.61	78.78	71.62	61.62	61.37	71.48	**68.25**†

Table 1: **Overall XGLUE Score for Iterative Pruning of XLM-R$_{\text{Base}}$ @ 31% Remaining Weights.**

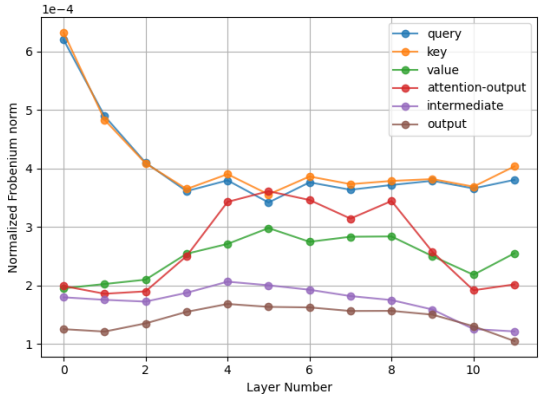

Figure 6: **Pruned Model Weight Norms Per Layer**

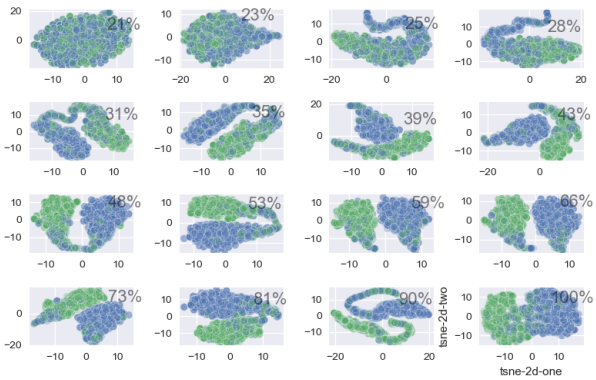

Figure 7: **Class Separability Between Class Representations At Each Iterative Pruning Step on PAWSX.**

chooses the majority of weights to prune from the layer type that has the smallest norm, leading to an information bottleneck, or layer collapse (Lee et al., 2018) for very high compression rates. This effect is due to layer normalization being applied after query, key and value (QKV) parameters, rescaling features such that weight magnitudes remain low. Hence, this motivates why we have focused on the application of *AlignReg* to layer-wise MBP. This is reflected in Figure 6 which shows the weight norm by layer type for each layer for MBP. We see that QKV weight values are distinctly higher than the remaining fully-connected layers (attention output layer, intermediate position-wise feed-forward layer and the blocks output layer), with the exception that the output attention layer norm becomes higher between layer 3-8. Lastly, we note that for the majority of tasks, the rate of performance drop for zero-shot test performance occurs close to 30% of remaining weights. This is consistent for all pruning methods. Figure 7 visualizes the class separability via a t-SNE plot of two principal components of the last hidden representation corresponding to the [CLS] token of an iteratively pruned XLM-R$_{\text{Base}}$ for PAWSX. Even from only two principal components of a single token input, we clearly see a change in class separability from 31% to 28% remaining weights, reflecting the lack of linear separation.

6 Conclusion

In this paper, we analysed iterative pruning in the zero-shot setting where a pretrained masked language model uses self-supervised learning on text from various languages but can only use a single language for downstream task fine-tuning. We find that some languages degrade in iterative pruning performance faster than others for some tasks (NER and XNLI) and propose a weight regularizer that biases the iteratively pruned model towards learning weight distributions close to the cross-lingually aligned pretrained state. This improves over well-established weight regularization methods for magnitude-based pruning in both the standard supervised learning setting and the zero-shot setting.

References

Johanni Brea, Berfin Simsek, Bernd Illing, and Wulfram Gerstner. 2019. Weight-space symmetry in deep networks gives rise to permutation saddles, connected by equal-loss valleys across the loss landscape. *arXiv preprint arXiv:1907.02911*.

Maxime Bucher, Stéphane Herbin, and Frédéric Jurie. 2017. Generating visual representations for zero-shot classification. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2666–2673.

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

Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. 2020. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. *arXiv preprint arXiv:2004.12651*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Xiaohan Ding, Guiguang Ding, Jungong Han, and Sheng Tang. 2018. Auto-balanced filter pruning for efficient convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Xin Dong, Shangyu Chen, and Sinno Jialin Pan. 2017. Learning to prune deep neural networks via layer-wise optimal brain surgeon. *arXiv preprint arXiv:1705.07565*.

G.H. Golub, Alan Hoffman, and G.W. Stewart. 1987. A generalization of the eckart-young-mirsky matrix approximation theorem. *Linear Algebra and its Applications*, 88-89:317–327.

Mitchell A Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing bert: Studying the effects of weight pruning on transfer learning. *arXiv preprint arXiv:2002.08307*.

S Han, H Mao, and WJ Dally. 2015a. Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint*.

Song Han, Jeff Pool, John Tran, and William J Dally. 2015b. Learning both weights and connections for efficient neural networks. *arXiv preprint arXiv:1506.02626*.

Babak Hassibi and David G Stork. 1993. *Second order derivatives for network pruning: Optimal brain surgeon*. Morgan Kaufmann.

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

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Steven A Janowsky. 1989. Pruning versus clipping in neural networks. *Physical Review A*, 39(12):6600.

Ehud D Karnin. 1990. A simple procedure for pruning back-propagation trained neural networks. *IEEE transactions on neural networks*, 1(2):239–242.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Vadim Lebedev and Victor Lempitsky. 2016. Fast convnets using group-wise brain damage. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2554–2564.

Yann LeCun, John S Denker, and Sara A Solla. 1990. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605.

Jaeho Lee, Sejun Park, Sangwoo Mo, Sungsoo Ahn, and Jinwoo Shin. 2020. Layer-adaptive sparsity for the magnitude-based pruning. In *International Conference on Learning Representations*.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. 2018. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, et al. 2020. Xglue: A new benchmark datasetfor cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. 2017. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744.

Christos Louizos, Max Welling, and Diederik P Kingma. 2017. Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2016. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*.

Michael C Mozer and Paul Smolensky. 1989. Skeletonization: A technique for trimming the fat from a network via relevance assessment. In *Advances in neural information processing systems*, pages 107–115.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal dependencies v2: An evergrowing multilingual treebank collection. *arXiv preprint arXiv:2004.10643*.

Sejun Park, Jaeho Lee, Sangwoo Mo, and Jinwoo Shin. 2020. Lookahead: A far-sighted alternative of magnitude-based pruning. *arXiv preprint arXiv:2002.04809*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

Russell Reed. 1993. Pruning algorithms-a survey. *IEEE transactions on Neural Networks*, 4(5):740–747.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Victor Sanh, Thomas Wolf, and Alexander M Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. *arXiv preprint arXiv:2005.07683*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Sidak Pal Singh and Dan Alistarh. 2020. Woodfisher: Efficient second-order approximations for model compression. *arXiv preprint arXiv:2004.14340*.

Slawomir W Stepniewski and Andy J Keane. 1997. Pruning backpropagation neural networks using modern stochastic optimisation techniques. *Neural Computing & Applications*, 5(2):76–98.

Xu Sun, Xuancheng Ren, Shuming Ma, and Houfeng Wang. 2017. meprop: Sparsified back propagation for accelerated deep learning with reduced overfitting. In *International Conference on Machine Learning*, pages 3299–3308. PMLR.

Chaoqi Wang, Roger Grosse, Sanja Fidler, and Guodong Zhang. 2019. Eigendamage: Structured pruning in the kronecker-factored eigenbasis. In *International Conference on Machine Learning*, pages 6566–6575. PMLR.

Huan Wang, Can Qin, Yulun Zhang, and Yun Fu. 2020. Neural pruning via growing regularization. *arXiv preprint arXiv:2012.09243*.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. *arXiv preprint arXiv:1608.03665*.

Shijie Wu, Alexis Conneau, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Emerging cross-lingual structure in pretrained language models. *arXiv preprint arXiv:1911.01464*.

Jianbo Ye, Xin Lu, Zhe Lin, and James Z Wang. 2018. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. *arXiv preprint arXiv:1802.00124*.

Ming Yuan and Yi Lin. 2006. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67.

Michael Zhu and Suyog Gupta. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*.