Pruning By Explaining Revisited: Optimizing Attribution Methods to Prune CNNs and Transformers

Sayed Mohammad Vakilzadeh Hatefi[1], Maximilian Dreyer[1], Reduan Achtibat[1], Thomas Wiegand[1,2,3], Wojciech Samek[1,2,3,†], and Sebastian Lapuschkin[1,†]

[1] Fraunhofer Heinrich-Hertz-Institute, 10587 Berlin, Germany
[2] Technische Universität Berlin, 10587 Berlin, Germany
[3] BIFOLD – Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany
[†] corresponding authors:
{wojciech.samek,sebastian.lapuschkin}@hhi.fraunhofer.de

Abstract. To solve ever more complex problems, Deep Neural Networks are scaled to billions of parameters, leading to huge computational costs. An effective approach to reduce computational requirements and increase efficiency is to prune unnecessary components of these often over-parameterized networks. Previous work has shown that attribution methods from the field of eXplainable AI serve as effective means to extract and prune the least relevant network components in a few-shot fashion. We extend the current state by proposing to explicitly optimize hyperparameters of attribution methods for the task of pruning, and further include transformer-based networks in our analysis. Our approach yields higher model compression rates of large transformer and convolutional architectures (VGG, ResNet, ViT) compared to previous works, while still attaining high performance on ImageNet classification tasks. Here, our experiments indicate that transformers have a higher degree of over-parameterization compared to convolutional neural networks. Code is available at `https://github.com/erfanhatefi/Pruning-by-eXplaining-in-PyTorch`.

Keywords: Explainable AI · Pruning · Attribution Optimization

1 Introduction

In recent years, Deep Neural Networks (DNNs) have been growing larger increasingly, demanding ever more computational resources and memory. To address these challenges, several efficient architectures, such as MobileNet [22] or EfficientFormer [28], have been proposed to reduce computational costs. However, the gain in efficiency comes at the cost of performance, as inherently efficient architectures struggle to keep pace with the recent surge in high-performing but costly transformer models.
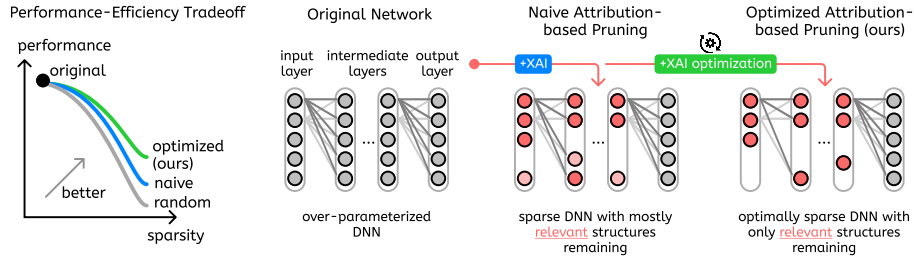
Fig. 1: We propose a pruning framework based on optimizing attribution methods from the field of eXplainable Artificial Intelligence (XAI). Compared to random pruning, pruning the least relevant structures first ("relevant" according to an XAI attribution method of choice, and indicated by *red color*), has been shown to result in an improved performance-sparsity tradeoff (simplified illustration depicted). By optimizing attribution methods specifically for pruning, we can reduce the tradeoff even further.

One widely adopted approach within the community of efficient Deep Learning is quantization [17,49], which involves compressing model parameters or features by reducing the number of bits used for representation, potentially followed by re-training to regain lost model performance. Another effective approach is model sparsification, commonly referred to as pruning, where irrelevant structures within the model are removed to reduce complexity and improve efficiency. It is to note, that quantization and pruning can be applied in tandem [25]. In this work, unlike previous studies that mandate post-retraining in their framework [14,35], we focus on pruning without additional training.

The key challenge in pruning (without sequential re-training) is the identification of structures within the model that can be removed without adversely affecting its performance. Earlier works have shown that structures may be chosen according to, *e.g.*, parameter magnitudes [18,27], functional redundancy [16], sensitivity [31,47], or importance in the decision-making process [6,48]. Especially the latter approach has been promising, utilizing tools from the seemingly unrelated field of XAI that originally seeks to explain the model reasoning process to human stakeholders.

Backpropagation-based attribution methods, particularly Layer-wise Relevance Propagation (LRP) [4,32,33], play a crucial role in this context. LRP works by tracing the internal reasoning of the model, assigning contribution scores to *latent* neurons, which highlight their importance to the final prediction. These scores provide a quantitative basis for identifying (ir-)relevant structures and subgraphs within the model that are (un-)essential for making accurate predictions, thus guiding the pruning process.

While prior attribution-based pruning [48] achieved remarkable results, it is restricted to heuristically chosen LRP hyperparameters and specific Convolutional Neural Networks (CNNs).

Following up upon the work [34] that optimized LRP w.r.t. specific explainability metrics, we argue that LRP or any other tunable attribution method

can be optimized specifically for the goal of more effective pruning (see Fig. 1). In addition, recent works [2, 3] have extended LRP to transformer-based architectures, allowing us now to also apply attribution-based pruning to the Vision Transformer (ViT) architecture.

In this work, we revisit attribution-based model sparsification, and extend the current state-of-the-art by

1. proposing a novel pruning framework based on optimizing attribution method hyperparameters to achieve higher sparsification rates.
2. incorporating transformer-based architectures, such as ViTs, to our framework by using recently developed attribution methods.
3. discussing differences between CNNs and ViTs in terms of pruning and over-parameterization.
4. revealing that attributions optimized for explanations are not necessarily the best for pruning.

2   Related Work

In the following, we introduce related works in the field of XAI, efficient Deep Learning and the intersection between both.

Explainability and Local Feature Attribution Methods. Research in local explainability led to a plethora of methods (*e.g.*, [29, 36]), commonly resulting in local feature attributions quantifying the importance of input features in the decision-making process. These attributions are often shown in the form of heatmaps in the vision domain. Notably, methods based on (modified) gradients, backpropagate attributions from the output to the input through the network, conveniently offering attributions of *all* latent components and neurons in a single backward pass [41, 43]. Gradient-based attribution methods, however, can suffer from noisy gradients, rendering them unreliable for deep architectures [5]. Prominently, LRP [4, 32] introduces a set of different rules (with hyperparameters) that allow to reduce the noise level. In fact, as shown in [34], attribution methods such as LRP can be optimized for certain XAI criteria, *e.g.*, faithfulness or complexity [21, 32]. We follow up on this observation, and specifically optimize XAI w.r.t. the task of pruning. That is, we add Neural Network (NN) pruning as an XAI evaluation criterion to optimize for.

Pruning of Deep Neural Networks. For pruning CNNs, either individual (kernel) weights or whole structures, *e.g.*, filters of convolution layers or neurons can be recognized as candidates for pruning [20]. For transformer architectures, such structures include heads of attention modules or linear layers inside transformer blocks [26, 45]. In order to prune such structures, several criteria have been proposed to indicate which components are best suited to be removed, retaining performance as best as possible. The work of [18] suggests pruning parameters based on weight magnitudes, offering a computationally free criterion. Alternatively, in [10], the authors propose to prune neurons based on their activation

patterns. However, recent work [6] has shown that weight or activation magnitudes do not necessarily reflect a component's contribution during the inference process, *e.g.*, also a small weight has the potential to be very relevant for the prediction of a class.

Explainability For Efficient Deep Learning. The work of [48] introduces a novel pruning criterion based on XAI, and proposes to use latent relevance values extracted from the attribution method of LRP [4, 33] to assign importance scores to network structures. By taking into account how structures are used during the inference process, the work of [48] can effectively improve the efficiency-performance tradeoff. Later, the works of [6, 42] also illustrate the value of XAI methods for network quantization. In all these works, heuristically chosen hyperparameters for LRP have been used, which overlooks the potential for specific optimization to the task of pruning. Notably, LRP is model-specific and thus restricted to compatible architectures. In this work, we also include recent LRP extensions to transformer architectures, and observe, that a specific LRP rule commonly result in high pruning performances.

3   Methods

This work proposes a framework for pruning DNNs using attribution methods from the field of XAI with hyperparameters specifically optimized for sparsification. We begin with presenting our method in the form of a general XAI-based pruning principle in Sec. 3.1, followed by introducing LRP attributions and corresponding hyperparameters suitable for optimization, in Sec. 3.2 and Sec. 3.3, respectively. Lastly, Sec. 3.4 describes our optimization methodology.

3.1   Attribution-based Pruning

For our structured pruning framework, we view a DNN as a collection of p (interlinked) components $\Psi = \{\psi_1, \ldots, \psi_p\}$, that can correspond to, *e.g.*, whole layers, (groups of) neurons, convolutional filters or attention heads. We further assume access to an attribution method that generates attribution scores (relevance scores) $R_{\psi_k}(x_i)$ of component $\psi_k \in \Psi$ for the prediction of a sample x_i. The overall relevance of ψ_k is then estimated through the mean relevance over a set of reference samples $\mathcal{X}_{\text{ref}} = \{x_1, x_2, \ldots, x_{n_{\text{ref}}}\}$ as

$$\bar{R}_{\psi_k} = \frac{1}{n_{\text{ref}}} \sum_{i=1}^{n_{\text{ref}}} R_{\psi_k}(x_i) \tag{1}$$

We collect relevance scores for all components via the set $\mathcal{R}$, given as

$$\mathcal{R} = \{\bar{R}_{\psi_1}, \bar{R}_{\psi_2}, \ldots, \bar{R}_{\psi_p}\} \tag{2}$$

which, in turn, allows to define a pruning order for the model components. Specifically, the indices c for the components to be pruned up to the q-th place are given by
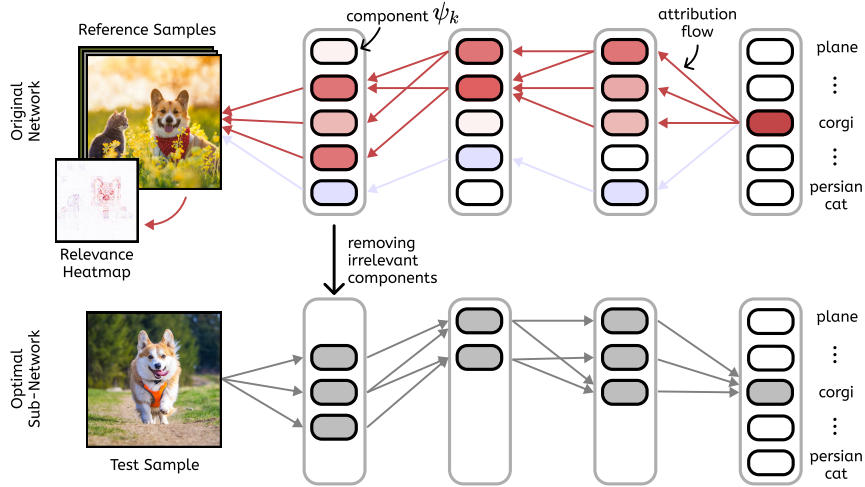
Fig. 2: Attribution-based pruning workflow: Firstly, the relevant model structures are identified by explaining a set of reference samples. The attribution method of choice (here LRP) highlights the components and paths in the network which positively and negatively contribute to the decision-making. Positive and negative relevances are indicated by red and blue color respectively, and components with white color indicate zero or low relevance. Removing structures that receive the least relevance results in a sparser subnetwork, which performs significantly better than after random pruning. Notably, relevances can be computed w.r.t. a subset of output classes (*e.g.*, "corgi" only), resulting in a subnetwork specifically designed to perform the restricted task. Credit: Nataba/iStock.

$$\{c\}_q = \mathrm{argsort}(\mathcal{R})_{1,2,\dots,q} \tag{3}$$

resulting in the set of least relevant components. Finally, the q least relevant components are pruned by removing or masking the components from the computational graph as

$$\forall \psi_i \in \Psi : \psi_i \mapsto (1 - \mathbf{1}_{i \in \{c\}_q})\psi_i \tag{4}$$

where $\mathbf{1}$ is an indicator function with condition $i \in \{c\}_q$. The whole attribution-based pruning workflow is depicted in Fig. 2. Notably, previous work [48] demonstrates that LRP is an effective method for attributing latent structures. LRP further offers several hyperparameters to tune, which makes it a versatile choice for our framework.

3.2 Layer-wise Relevance Propagation

Layer-wise Relevance Propagation [4, 33] is a rule-based backpropagation algorithm that was designed as a tool for interpreting non-linear learning models

by assigning attribution scores, called "relevances", to network units proportionally to their contribution to the final prediction value. Unlike other gradient- or perturbation-based methods, LRP treats a neural network as a layered directed acyclic graph with L layers and input x:

$$f(x) = f^L \circ \cdots \circ f^l \circ f^{l-1} \circ \cdots \circ f^1(x) \tag{5}$$

Beginning with an initial relevance score R_j^L at output j of layer f^L (usually set as f_j^L for an output of choice j), the score is layer-by-layer redistributed through all latent structures to its input variables depending on the contribution from the these units to the output value:

Given a layer, we consider its pre-activations z_{ij} mapping inputs i to outputs j and their aggregations $z_j = \sum_i z_{ij}$. Commonly in linear layers such a computation is given with $z_{ij} = a_i w_{ij}$, where w_{ij} are its weight parameters and a_i the activation of neuron i.

Then, LRP distributes relevance quantities R_j^l received from upper layers towards lower layers proportionally to the relative contributions of z_{ij} to z_j, i.e.,

$$R_{i \leftarrow j}^{(l-1,l)} = \frac{z_{ij}}{z_j} R_j^l \tag{6}$$

In other words, the relevance message $R_{i \leftarrow j}^{(l-1,l)}$ quantifies the contribution of neuron i at layer $l-1$, to the activation of neuron j at layer l.

To obtain the contribution of neuron i to all upper layer neurons j, all incoming relevance messages $R_{i \leftarrow j}^{(l-1,l)}$ are losslessly aggregated as

$$R_i^{l-1} = \sum_j R_{i \leftarrow j}^{(l-1,l)} \tag{7}$$

This process ensures relevance conservation between adjacent layers:

$$\sum_i R_i^{l-1} = \sum_{i,j} R_{i \leftarrow j}^{(l-1,l)} = \sum_j R_j^l \tag{8}$$

which guarantees that the sum of all relevance in each layer stays the same.

When a group of neurons performs the same task as within a convolutional channel or attention head, it is beneficial to aggregate the total relevance of the entire group into a single relevance score. This aggregation process helps in simplifying the analysis and interpretation of the model's behavior by focusing on the collective relevance of each convolutional channel or attention head, rather than examining individual neurons. Further discussions of component-wise aggregation are given in Appendix B.

3.3   Tuneable Hyperparameters of LRP

Within the LRP framework, various rules (as detailed in Appendix A.2) have been proposed to tune relevance computation for higher explainability.

Further, several works have shown that applying different rules to variable parts of a network (referred to as a "composite") [2, 24, 32] enhances faithfulness and robustness of explanations. We follow up on this idea, and perform a large-scale analysis over a variety of rules for different network parts. Inspired by the tuning configurations of [34], we split a network into four parts:

1. the final classification layers, denoted as Fully-Connected Layers (FCL),
2. the last 25% of hidden layers before the FCL, denoted as High-Level hidden Layers (HLL),
3. the first 25% of hidden layers, denoted as Low-Level hidden Layers (LLL),
4. and the remaining hidden layers between HLL and LLL, denoted as Mid-Level hidden Layers (MLL)

For each group, a specific rule set can be chosen, which together ultimately form the LRP composite. We refrain from treating each individual layer as a separate group due to the increased computational expenses associated with this setting during hyperparameter search.

For transformer architectures, the works of [2, 3] proposed novel LRP rules to attribute softmax non-linearities in the attention modules (see Appendix A.2 for more details), encouraging us to also optimize LRP configurations for the softmax non-linearities, resulting in an additional hyperparameter choices here.

It is important to note, that relevance values can be positive or negative. Components with positive relevance values indicate that they significantly contribute towards the explained class prediction. Conversely, components with negative relevance values diminish the classification score, effectively speaking against the predicted class, as visible in Fig. 2 where "cat" features (and the respective model components) speak against the "corgi" prediction. Components with near zero relevance do not contribute to the inference process in any meaningful way. Thus, a question arises w.r.t. the optimal order of pruning: Should we start with negative, or near zero relevant components? To that end, we also add a hyperparameter indicating whether absolute relevance values are used as in Eq. (1), $i.e.$, $\bar{R}_{\psi_k} \mapsto |\bar{R}_{\psi_k}|$ for all $\psi_k \in \Psi$ if true. The work of [48] takes into account and computes only positive attributions by their LRP rule choice (see LRP-z^+ in Appendix A.2). Our experiments in Appendix F, however, reveal that the highest pruning rates are achieved by computing both positive and negative relevance signals, and then first pruning components with near zero relevance, as their minimal contribution in any direction ensures low impact on the overall model performance.

3.4   Hyperparameter Optimization Procedure

In order to optimize the hyperparameters of an attribution method, we first define an optimization objective C. As we perform our analysis for classification tasks, we measure the top-1 accuracy on the validation dataset. In principle, a different performance criterion can be chosen here. Concretely, we measure model performance for different pruning rates $\text{PR}_i = \frac{1}{m}i$ as given after step

$i \in \{0, \ldots m-1\}$ of in total m steps (excluding a 100% rate). After sequentially increasing the pruning rate, and plotting performance against sparsity, we receive a curve that indicates the sparsity-accuracy tradeoff as, *e.g.*, shown in Fig. 1 (*left*). We therefore propose to measure the resulting area under the curve A_{PR} as given by

$$A_{\mathrm{PR}} = \frac{1}{m} \sum_{i=0}^{m-1} C(f_{\Psi_{\mathrm{PR}_i}(\theta)}) \tag{9}$$

where the performance as given by C depends on the network f and its pruned parameters $\Psi_{\mathrm{PR}_i}(\theta)$ after pruning step i using the hyperparameter setting θ for the attribution method. Ultimately, our objective will be the maximization of A_{PR} w.r.t. θ. Hyperparameter search is performed via grid search and Bayesian optimization techniques, as detailed in Appendix D.

4   Experiments

We begin our experiments with exploring the over-parameterization problem of DNNs in Sec. 4.1. This is followed by our results on finding the best LRP hyperparameters for pruning CNNs and ViTs in Sec. 4.2 and Sec. 4.3, respectively. Lastly, we compare the effect of pruning using ideal and random attributions in Sec. 4.4 by evaluating how explanation heatmaps change after pruning.

Experimental Setting In our experiments, we optimize LRP hyperparameters for pruning convolution filters of VGG-16 [40] (with and without BatchNorm [23] layers), ResNet-18 and ResNet-50 [19] architectures (with 4224, 4224, 4800 and 26560 filters overall, respectively), as well as linear layers and attention heads of the ViT-B-16 transformer [11] (with 46080 neurons and 144 heads). All models are pre-trained [30] and evaluated on the ImageNet dataset [9]. For hyperparameter optimization, we measure model performance for 20 pruning rates (from $0\,\%$ up to $95\,\%$) on the validation dataset. To compute latent attributions, a set of reference samples has been chosen from the training set of ImageNet (different set sizes are discussed later in Sec. 4.2 and 4.3).

4.1   How Over-Parameterized are Vision Models?

Training a large DNN from scratch to solve a specific task can be computationally expensive. A popular approach for saving training resources is to instead fine-tune a large pre-trained (foundation) model, which often requires fewer training epochs and provides high (or even higher) model generalization, especially in sparse data settings [8]. Notably, one of the effects that arise when solving for simple(r) tasks, is that we likely end up with an over-parameterized model as only a subset of very specialized latent features are necessary to solve the task, which makes pruning especially interesting in this case.

In fact, when pruning the ResNet-18 and VGG-16-BN models that were pre-trained to detect *all* of the 1000 ImageNet classes, parameter count can only
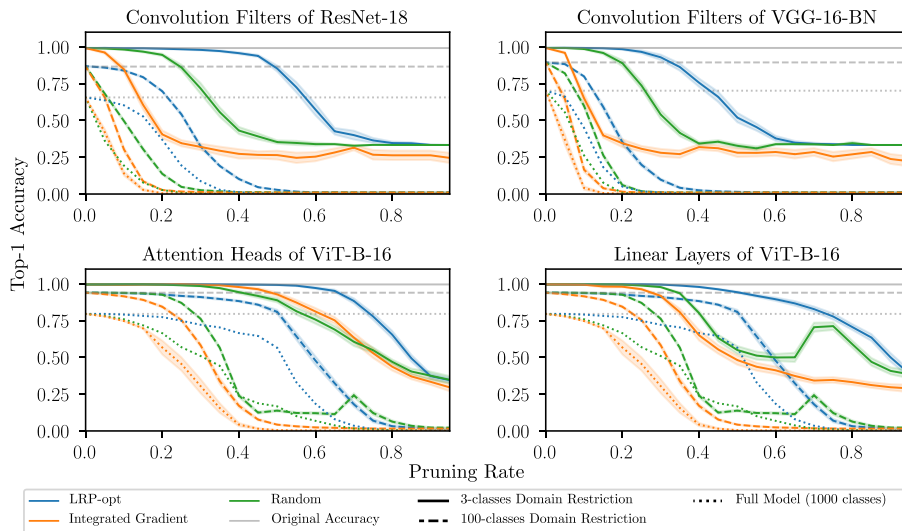
Fig. 3: Investigating over-parameterization in DNNs through attribution-based pruning (LRP-opt in *blue*, Integrated Gradient in *orange color*) and random pruning (*green color*). We compare pruning of all models w.r.t. different task difficulties, *i.e.*, to differentiate between 1000 (*dotted line*), 100 (*dashed line*) or three ImageNet classes (*solid line*). High performance for high sparsification rates indicates over-parameterization, *i.e.*, many network components are not important for the task. Compared to the ResNet-18 and VGG-16-BN CNNs (*top*), the ViT-B-16 transformer shows a higher degree of over-parameterization (*bottom*). Standard Error of Mean (SEM) is illustrated (*shaded area*) in the current and all other figures.

be reduced by a couple of percent without meaningful degradation of model performance, as shown in Fig. 3 (*top*). This indicates that ImageNet itself is a complex task, and most of the trained parameters are actually relevant. However, as the task becomes simpler (simulated by reducing the number of output classes to 100 or three in the evaluation), pruning rates can be increased to a much higher level without critical accuracy loss.

For baseline comparisons, we utilize random pruning, Integrated Gradient [43] and our optimized LRP attribution (LRP-opt) (see Sec. 4.2). We intentionally exclude naive weight magnitude or activation pruning since [48] has previously demonstrated that XAI-based methods outperform these traditional approaches by a large margin. Notably, especially for simpler tasks, attribution-based pruning outperforms random pruning, as attributions are output-specific and allow identifying the sub-graph that is relevant for the restricted task, as also illustrated in Fig. 2.

Compared to the previously discussed CNNs, the ViT-B-16 vision transformer seems to be more over-parameterized, as visible in Fig. 3 (*bottom*), and therefore easier to prune. Up to a pruning rate of 20 %, no meaningful accuracy loss is measured for detecting the 1000 ImageNet classes, irrespective of whether
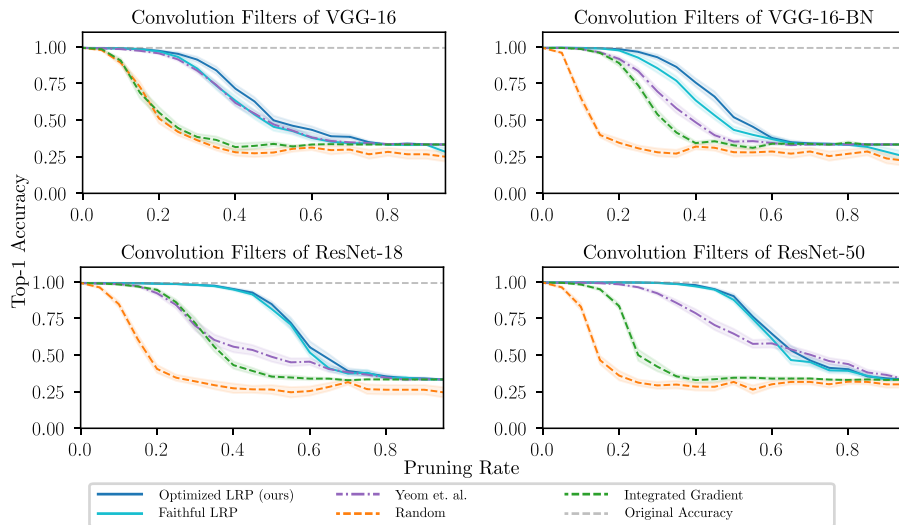
Fig. 4: Pruning CNNs models pre-trained on ImageNet (simplified task to detecting three classes), using ten reference samples per class. Results show a better sparsification-performance tradeoff for our optimized LRP composite compared to a heuristic (faithful) LRP composite, Yeom *et al.* [48] (details for each in Appendix F.1, Appendix A.5, and Appendix A.3) and random pruning.

attention heads or neurons in linear layers are pruned. This might be due to the fact, that the vision transformer consists of about 46,000 neurons, which is approximately ten times the number of convolutional filters of the CNNs. Also, when comparing the ResNet-50 and ResNet-18 (see Fig. A.1 for ResNet-50), the larger network shows higher pruning potential.

4.2   Finding the Optimal Attributions for CNNs

Motivated by the previous Sec. 4.1, we in the following simulate a setting with over-parameterized networks that allows us to measure more significant differences between methods. Specifically, we restrict the data domain to three ImageNet classes and evaluate pruning using 20 different random seeds. Experiments using toy models in the work of [48] showed that ten (or more) reference samples are already well suited for estimating the overall relevances of network components (as used in Eq. (1)). We validated their finding also for the ResNet-18 model, as depicted in Fig. A.2.

Whereas we begin with optimizing LRP for CNN pruning, we follow up with transformers in the next section. Regarding CNNs, we can strongly reduce the accuracy-sparsity tradeoff when pruning the convolution filters with our method based on optimizing LRP attributions compared to other baselines, *e.g.*, the heuristically chosen variant of LRP used by [48] (see Appendix A.3), as also illustrated in Fig. 4 and Sec. 4.2. Extensive hyperparameter search reveals that

Table 1: Results for pruning CNN models pre-trained on ImageNet. **Top-PR** demonstrates the highest pruning rate while keeping 95% of baseline accuracy. **F**, **IG**, and **R** in order indicate a faithful LRP composite, Integrated Gradients, and a random pruning baseline. Our results remark high scores for the area under the induced curve (A^{PR}) in pruning, overall improving the performance-sparsity tradeoff.

Models	A^{PR}					**Top-PR** $(\%) \pm 3\,(\%)$				
	Ours	**F**	**[48]**	**IG**	**R**	**Ours**	**F**	**[48]**	**IG**	**R**
VGG-16	**0.61**	0.58	0.58	0.43	0.4	**26**	23	21	7	6
VGG-16-BN	**0.61**	0.57	0.53	0.50	0.35	**28**	23	17	16	5
ResNet-18	**0.69**	0.68	0.57	0.53	0.37	**41**	40	18	20	5
ResNet-50	**0.72**	0.71	0.67	0.47	0.37	**45**	**45**	27	15	5

across different CNN architectures, simple hyperparameter settings exist that are well-suited for pruning in general. One such candidate is the LRP-ϵ rule (see Eq. (A.3)) applied for all convolutional layers. We refer to Appendix F.1 for a list of the best performing hyperparameter settings. Interestingly for CNNs, LRP composites known to lead to faithful explanations (in the suggested context of [4, 37]) optimized by [24], are also effective in pruning. One such composite (detailed in Appendix A.5) is also shown in Fig. 4 and denoted as "Faithful LRP". Moreover, our results indicate a larger amount of unused structures across architectures with shortcut connections, *i.e.*, the ResNet models. As intermediate features can be passed through the short-cuts, while layers have the potential to be not used at all.

It is further not surprising that faithful LRP composites w.r.t. *input* explanations also perform well in pruning *latent* structures. Commonly, faithfulness is measured by "deleting" or perturbing input features and measuring the effect on the model's prediction [21, 37]. A highly (or lowly) relevant feature is assumed to result in a high (or low) change in model output. As such, attribution-based pruning in combination with measuring the model performance resembles a faithfulness evaluation scheme in *latent* space. The more faithful the attributions, the smaller should be the degradation of model performance. However, later, in Sec. 4.3, we can also observe ineffective pruning with an attributor that is known to be faithful in input space.

It is to note that the LRP-ε rule is known to result in noisy (and not necessarily faithful [37]) input attributions for DNNs, but performs very well for attributing latent components, as visible in Appendix F.1. We hypothesize that this results from fewer noise in intermediate layers [5] and the fact that we aggregate attributions for each component (*e.g.*, channel) which further reduces noise, as also proposed for gradients for the GradCAM method [38]. High *latent* faithfulness with noisy input attributors (LRP-ε or gradient times input [39]) has also been observed in [12, 13].
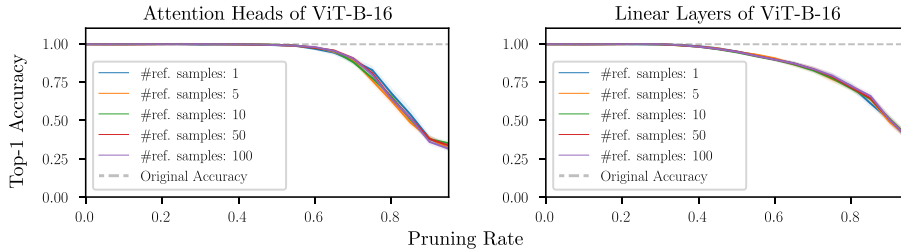
Fig. 5: Attribution-based pruning using a different number of reference samples (per class) to estimate the importance of attention heads (*left*) or neurons in linear layers (*right*) of the ViT-B-16. This experiment has been conducted for 20 different random seeds. For the propagation of LRP, LRP-ϵ has been set as our parameter for all layers (Sec. 3.3), and w.r.t. the attribution of softmax operations (Appendix A.4).
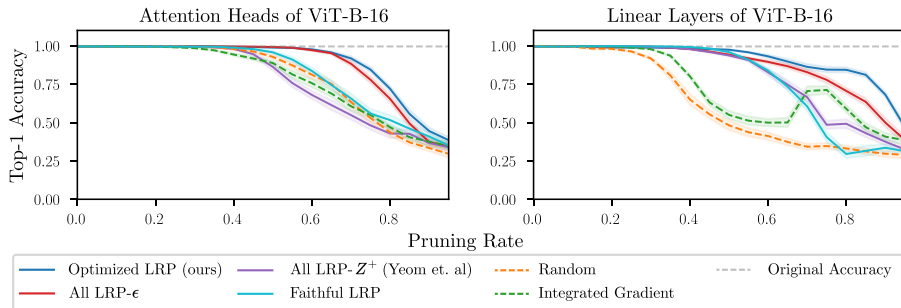

Fig. 6: Pruning a ViT model pre-trained on ImageNet (simplified task to detecting three classes), using ten reference samples per class. Results show a better sparsification-performance tradeoff for our optimized LRP composite compared to a heuristic (faithful) LRP composite, Yeom *et al.* [48] (details for each in Appendix F.1, Appendix A.5, and Appendix A.3, respectively), LRP-ϵ, and random pruning.

4.3   Finding the Optimal Attributions for Vision Transformer

Attention heads and linear layers are two structures of interest in the pruning literature for Vision Transformers [26,44]. As observed in Sec. 4.1, the ViT model shows a higher degree of over-parameterization, possibly due to the abundance of neurons in the linear layers, ultimately highlighting the possible value for pruning. We now re-investigate the number of reference samples required to robustly estimate the relevance of a component in the ViT-B-16 model. Interestingly, unlike CNNs, according to Fig. 5, there is no deviation in pruning reliability and stability by using different numbers of reference samples. Consequently, our experiments have been applied with exact same settings as in Sec. 4.2.

Our results from Fig. 6 again confirm an improvement of our optimized attribution-based pruning scheme (best LRP composites shown in Tab. A.1) compared to the previous work of [48]. Adopting the generally reliable composite of LRP-ϵ obtained from CNNs (as in Fig. 4 and Appendix F.1) is again a

Table 2: Results for pruning the ViT-B-16 pre-trained on ImageNet. **Top-PR** demonstrates the highest pruning rate while keeping 95% of baseline accuracy. **F**, **IG**, and **R** in order indicate a faithful LRP composite, Integrated Gradients, and a random pruning baseline. Our results remark high scores for the area under the induced curve (A^{PR}) in pruning, overall improving the performance-sparsity tradeoff.

Models	A^{PR}						**Top-PR** (%) $\pm$ 3 (%)					
	Ours	F	LRP-ϵ	[48]	IG	R	Ours	F	LRP-ϵ	[48]	IG	R
Linear Layers	**0.87**	0.75	0.83	0.77	0.70	0.59	**57**	51	49	48	33	26
Attention Heads	**0.85**	0.78	0.85	0.74	0.75	0.76	**66**	51	64	44	39	47

promising option. Nonetheless, unlike for CNNs, a recently proposed faithful LRP composite [2] designed for the ViT-B-16 model (detailed in Appendix A.5), is *not* ideal for pruning. This finding, and our experiment in Appendix F.2 show that optimizing an attributor for two different contexts of faithfulness (input or latent space/pruning) (Sec. 3.3) does not necessarily lead to an attributor that attributes faithfully in both input and latent space. The poorer performance of heuristically tuned LRP from [48] (Appendix A.3) compared to random when pruning attention heads, underscores both the greater challenge of pruning this structure and the significance of our proposed optimization procedure.

4.4   How Pruning Affects Model Explanations

In addition to keeping track of the model performance, the field of XAI encourages us in the following to investigate the model behavior by observing explanation heatmaps. Concretely, we expect in the ideal pruning scheme, that heatmaps change as late as possible when increasing the pruning rate. This reflects that the task-relevant components are retained as long as possible.

   As an illustrative example, we prune the attention heads of the ViT-B-16 model pre-trained on ImageNet with the aim to predict ImageNet corgi classes ("Pembroke Welsh" and "Cardigan Welsh") as shown in Fig. 7. As a quantitative measure, we compute the cosine similarity between the original heatmap (using the recently proposed composite of [1]) and the heatmap of the pruned model for different pruning rates over the validation set.

   On the one hand, we can see that random pruning or unoptimized attribution-based pruning leads to a much earlier change in heatmaps compared to our optimized LRP-based approach. The heatmaps indicate that irrelevant features are removed first as the pruning rate is increased (*e.g.*, "cat" features in Fig. 7 get disregarded before corgi-related features in the process). However, a random pruning approach might wrongly omit an important structure first, as heatmap changes of "mouth" and "ear" features can be seen in Fig. 7. On the other hand, as can be expected, the change in heatmap similarity highly correlates with the model confidence, resulting in a Pearson correlation coefficient of 0.99.
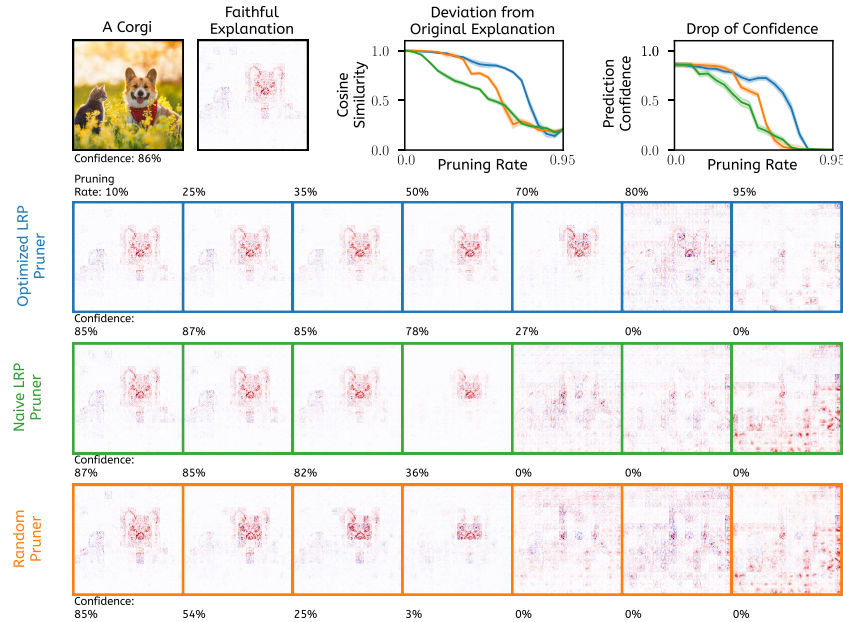
Fig. 7: Pruning a ViT with the aim to retain a high accuracy for detecting ImageNet corgi classes ("Pembroke Welsh" and "Cardigan Welsh"). We show, how explanation heatmaps for a corgi prediction change when increasing the pruning rate (*bottom*). Initially, most positive relevance lies on the corgi's head, with few negative relevance on a cat sitting next to the dog. As expected, when increasing the pruning rate using our optimized LRP composite, the background features disappear before corgi-related features are perturbed. When naively using the composite proposed in [48], or performing random pruning, heatmaps change much earlier (and more randomly), indicating model degradation. In fact, we can measure a high correlation of 0.99 between heatmap change and confidence loss (*top right*). Surprisingly, the unoptimized LRP composite performs even worse than random pruning, stressing the need to optimize attribution methods before applying them heuristically.

5   Conclusion

In this work, we propose a general framework for post-hoc DNN pruning that is based on using and optimizing attribution-based methods from the field of eXplainable Artificial Intelligence for more effective pruning. For our framework, the method of LRP is well-suited by offering several hyperparameters to tune attributions. When applying our framework, we can strongly reduce the performance-sparsity tradeoff of CNNs and especially ViTs compared to previously established approaches. Vision transformers are on the one hand more sensitive towards hyperparameters, and also show higher over-parameterization. Overall, using local XAI methods for pruning irrelevant model components demonstrates high potential in our experiments.

Acknowledgements

References

1. Achtibat, R., Dreyer, M., Eisenbraun, I., Bosse, S., Wiegand, T., Samek, W., Lapuschkin, S.: From attribution maps to human-understandable explanations through concept relevance propagation. Nature Machine Intelligence **5**(9), 1006–1019 (2023)
2. Achtibat, R., Hatefi, S.M.V., Dreyer, M., Jain, A., Wiegand, T., Lapuschkin, S., Samek, W.: AttnLRP: Attention-aware layer-wise relevance propagation for transformers. In: Proceedings of the 41st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 235, pp. 135–168. PMLR (21–27 Jul 2024)
3. Ali, A., Schnake, T., Eberle, O., Montavon, G., Müller, K.R., Wolf, L.: Xai for transformers: Better explanations through conservative propagation. In: International Conference on Machine Learning. pp. 435–451. PMLR (2022)
4. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PloS one **10**(7), e0130140 (2015)
5. Balduzzi, D., Frean, M., Leary, L., Lewis, J., Ma, K.W.D., McWilliams, B.: The shattered gradients problem: If resnets are the answer, then what is the question? In: International Conference on Machine Learning. pp. 342–350. PMLR (2017)
6. Becking, D., Dreyer, M., Samek, W., Müller, K., Lapuschkin, S.: Ecq x: explainability-driven quantization for low-bit and sparse dnns. In: International Workshop on Extending Explainable AI Beyond Deep Models and Classifiers. pp. 271–296. Springer (2020)
7. Blücher, S., Vielhaben, J., Strodthoff, N.: Decoupling pixel flipping and occlusion strategy for consistent xai benchmarks (2024)
8. Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., Bohg, J., Bosselut, A., Brunskill, E., et al.: On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258 (2021)
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
10. Dong, J., Zheng, H., Lian, L.: Activation-based weight significance criterion for pruning deep neural networks. In: Image and Graphics: 9th International Conference, ICIG 2017, Shanghai, China, September 13-15, 2017, Revised Selected Papers, Part II 9. pp. 62–73. Springer (2017)
11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2020)

12. Dreyer, M., Achtibat, R., Wiegand, T., Samek, W., Lapuschkin, S.: Revealing hidden context bias in segmentation and object detection through concept-specific explanations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 3828–3838 (2023)
13. Fel, T., Boutin, V., Béthune, L., Cadène, R., Moayeri, M., Andéol, L., Chalvidal, M., Serre, T.: A holistic approach to unifying automatic concept extraction and concept importance estimation. Advances in Neural Information Processing Systems **36** (2024)
14. Frankle, J., Carbin, M.: The lottery ticket hypothesis: Finding sparse, trainable neural networks. arXiv preprint arXiv:1803.03635 (2018)
15. Frazier, P.I.: A tutorial on bayesian optimization. arXiv preprint arXiv:1807.02811 (2018)
16. Geng, L., Niu, B.: Pruning convolutional neural networks via filter similarity analysis. Machine Learning **111**(9), 3161–3180 (2022)
17. Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M.W., Keutzer, K.: A survey of quantization methods for efficient neural network inference. In: Low-Power Computer Vision, pp. 291–326. Chapman and Hall/CRC (2022)
18. Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. Advances in neural information processing systems **28** (2015)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
20. He, Y., Kang, G., Dong, X., Fu, Y., Yang, Y.: Soft filter pruning for accelerating deep convolutional neural networks. In: IJCAI International Joint Conference on Artificial Intelligence (2018)
21. Hedström, A., Weber, L., Krakowczyk, D., Bareeva, D., Motzkus, F., Samek, W., Lapuschkin, S., Höhne, M.M.M.: Quantus: An explainable ai toolkit for responsible evaluation of neural network explanations and beyond. Journal of Machine Learning Research **24**(34), 1–11 (2023)
22. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
23. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. pmlr (2015)
24. Kohlbrenner, M., Bauer, A., Nakajima, S., Binder, A., Samek, W., Lapuschkin, S.: Towards best practice in explaining neural network decisions with lrp. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp. 1–7. IEEE (2020)
25. Kuzmin, A., Nagel, M., Van Baalen, M., Behboodi, A., Blankevoort, T.: Pruning vs quantization: which is better? Advances in neural information processing systems **36** (2024)
26. Lagunas, F., Charlaix, E., Sanh, V., Rush, A.M.: Block pruning for faster transformers. arXiv preprint arXiv:2109.04838 (2021)
27. Lee, J., Park, S., Mo, S., Ahn, S., Shin, J.: Layer-adaptive sparsity for the magnitude-based pruning. In: 9th International Conference on Learning Representations, ICLR 2021 (2021)
28. Li, Y., Yuan, G., Wen, Y., Hu, J., Evangelidis, G., Tulyakov, S., Wang, Y., Ren, J.: Efficientformer: Vision transformers at mobilenet speed. Advances in Neural Information Processing Systems **35**, 12934–12949 (2022)

29. Lundberg, S.M., Lee, S.: A unified approach to interpreting model predictions. In: Advances in Neural Information Processing Systems 30. pp. 4765–4774 (2017)

30. Marcel, S., Rodriguez, Y.: Torchvision the machine-vision package of torch. In: Proceedings of the 18th ACM international conference on Multimedia. pp. 1485–1488 (2010)

31. Molchanov, P., Tyree, S., Karras, T., Aila, T., Kautz, J.: Pruning convolutional neural networks for resource efficient inference. arXiv preprint arXiv:1611.06440 (2016)

32. Montavon, G., Binder, A., Lapuschkin, S., Samek, W., Müller, K.R.: Layer-wise relevance propagation: an overview. Explainable AI: interpreting, explaining and visualizing deep learning pp. 193–209 (2019)

33. Montavon, G., Lapuschkin, S., Binder, A., Samek, W., Müller, K.R.: Explaining nonlinear classification decisions with deep taylor decomposition. Pattern recognition **65**, 211–222 (2017)

34. Pahde, F., Yolcu, G.Ü., Binder, A., Samek, W., Lapuschkin, S.: Optimizing explanations by network canonization and hyperparameter search. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3818–3827 (2023)

35. Peste, A., Iofinova, E., Vladu, A., Alistarh, D.: Ac/dc: Alternating compressed/decompressed training of deep neural networks. Advances in neural information processing systems **34**, 8557–8570 (2021)

36. Ribeiro, M.T., Singh, S., Guestrin, C.: " why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 1135–1144 (2016)

37. Samek, W., Binder, A., Montavon, G., Lapuschkin, S., Müller, K.R.: Evaluating the visualization of what a deep neural network has learned. IEEE Transactions on Neural Networks and Learning Systems **28**(11), 2660–2673 (2017)

38. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Gradcam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision. pp. 618–626 (2017)

39. Shrikumar, A., Greenside, P., Kundaje, A.: Learning important features through propagating activation differences. In: International Conference on Machine Learning. pp. 3145–3153. PMLR (2017)

40. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations (ICLR 2015). Computational and Biological Learning Society (2015)

41. Smilkov, D., Thorat, N., Kim, B., Viégas, F., Wattenberg, M.: Smoothgrad: removing noise by adding noise. arXiv preprint arXiv:1706.03825 (2017)

42. Soroush, K., Raji, M., Ghavami, B.: Compressing deep neural networks using explainable ai. In: 2023 13th International Conference on Computer and Knowledge Engineering (ICCKE). pp. 636–641. IEEE (2023)

43. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: International Conference on Machine Learning. pp. 3319–3328. PMLR (2017)

44. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in Neural Information Processing Systems **30** (2017)

45. Voita, E., Talbot, D., Moiseev, F., Sennrich, R., Titov, I.: Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. arXiv preprint arXiv:1905.09418 (2019)

46. Williams, C., Rasmussen, C.: Gaussian processes for regression. Advances in neural information processing systems **8** (1995)
47. Yang, C., Liu, H.: Channel pruning based on convolutional neural network sensitivity. Neurocomputing **507**, 97–106 (2022)
48. Yeom, S.K., Seegerer, P., Lapuschkin, S., Binder, A., Wiedemann, S., Müller, K.R., Samek, W.: Pruning by explaining: A novel criterion for deep neural network pruning. Pattern Recognition **115**, 107899 (2021)
49. Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., Zou, Y.: Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. arXiv preprint arXiv:1606.06160 (2016)

Appendix

A   Attribution Methods

As it has been discussed in Sec. 3.1, backpropagation methods are suitable for our proposed framework. We focus mostly on LRP and it will be compared with Integrated Gradient as a commonly used gradient-based explainer.

A.1   Integrated Gradient

As an improvement to pure gradient, Integrated Gradient [43] interpolates input x for m steps and computes their gradients sequentially.

$$\text{IG}(x) = (x - x') \int_{\alpha=0}^{1} \frac{\partial f_j(x' + \alpha \times (x - x'))}{\partial x} dx \tag{A.1}$$

$$\approx (x - x') \sum_{k=1}^{m} \frac{\partial f_j(x' + \frac{k}{m} \times (x - x'))}{\partial x} \times \frac{1}{m} \tag{A.2}$$

Unlike LRP, there is no parameter to be tuned in this method. m from equation Eq. (A.1) only serves as an approximation factor, with higher steps indicating more precise computation of the integral. For the experiments conducted in this paper, we interpolate each input for 20 steps.

A.2   Variants of LRP

In addition to the vanilla rule of LRP demonstrated in 6, [4,32] later on proposed different rules as an extension of vanilla rule designed to serve different purposes. Most common extension of LRP will be discussed based on the layer type they target.

A.3   Tackling Linear and Convolution Layers

As a side note, in the propagation process of LRP, linear layers of DNNs such as fully connected, and convolution layers, are treated similarly.

LRP-ϵ The most fundamental problem of rule 6 causing computational instability, is division by zero which takes place when a neuron is not activated at all ($z_j = 0$). LRP-ϵ with $\epsilon \in \mathbb{R}$ as a stabilizer parameter, was proposed to tackle this problem so that the denominator never reaches zero:

$$R_{i \leftarrow j} = \frac{z_{ij}}{z_j + \epsilon \text{sign}(z_j)} R_j \tag{A.3}$$

As a side note, $\text{sign}(0) = 1$. Although the value of ϵ can be tuned, we set it to $1e - 6$ in every use-case.

LRP-$\alpha\beta$ As described in figure Fig. 2 and Sec. 3.3, relevances either have a positive or negative sign, indicating whether they have contributed positively or negatively in favor of the decision-making. Depending on the use-case, we might be interested in having a higher emphasis on positive relevances rather than the negative ones or vice versa. LRP-$\alpha\beta$ rule with $\alpha + \beta = 1$, lets us apply modifications in that regard.

$$R_{i\leftarrow j} = \left(\alpha\frac{z_{ij}^+}{z_j^+} + \beta\frac{z_{ij}^-}{z_j^-}\right)R_j \tag{A.4}$$

LRP-z^+ An extreme case for the above formulation when it is required to ignore all negative relevance, is the combination of $\alpha = 1$ and $\beta = 0$, which is referred to as LRP-z^+ used by [48].

LRP-γ Another derivation, with the same purpose of LRP-$\alpha\beta$ to control over the distribution of positive and negative relevance separately, has this form:

$$R_{i\leftarrow j} = \frac{z_{ij} + \gamma z_{ij}^+}{z_j + \gamma\sum_k z_{kj}^+}R_j \tag{A.5}$$

Evidently, in case of $\gamma = \infty$, LRP-γ will be equivalent to LRP-z^+. The value of γ can also be tuned in the optimization process. However, to prevent further computational costs unlike results from [2] in Appendix A.5, we refrain from tuning it.

A.4   Tackling Non-linearities

Activation Functions In LRP framework, a non-linear activation function σ (*e.g.*, ReLU), with $\sigma_i(z_i) = a_i$ is handled similar to Eq. (6), where $z_{ij} = \delta_{ij}z_i$ and δ represents the Kronecker delta.

Attention and Softmax on Transformers The key component of the attention module [44], consists of two equations:

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\cdot\mathbf{K}^T}{\sqrt{d_k}}\right) \tag{A.6}$$

$$\mathbf{O} = \mathbf{A}\cdot\mathbf{V} \tag{A.7}$$

where $(\cdot)$ denotes matrix multiplication, $\mathbf{K} \in \mathbb{R}^{b\times s_k\times d_k}$ is the key matrix, $\mathbf{Q} \in \mathbb{R}^{b\times s_q\times d_k}$ is the queries matrix, $\mathbf{V} \in \mathbb{R}^{b\times s_k\times d_v}$ the values matrix, and $\mathbf{O} \in \mathbb{R}^{b\times s_k\times d_v}$ is the final output of the attention mechanism. b is the batch dimension including the number of heads, and d_k, d_v indicate the embedding dimensions, and s_q, s_k are the number of query and key/value tokens.

CP-LRP In the extension of LRP from [3], it was proposed to regard the attention matrix (softmax output) $\mathbf{A}$ in Eq. (A.7) as constant, attributing relevance solely through the value path by stopping the relevance flow through the softmax. Consequently, the matrix multiplication Eq. (A.7) can be treated with the ε-rule Eq. (A.3).

AttnLRP The recent work of [2] extends LRP to Large Language Models and Vision Transformers by proposing to linearize the softmax function in Eq. (A.6) with a Deep Taylor Decomposition [33] which leads to substantially improved attributions in the natural language domain. For that, the softmax non-linearity Eq. (A.6) is attributed as follows:

$$R_i^l = x_i(R_i^{l+1} - s_i \sum_j R_j^{l+1}) \tag{A.8}$$

where s_i denotes the i-th output of the softmax. In addition, the matrix multiplication Eq. (A.7) is attributed with

$$R_{ji}^{l-1}(\mathbf{A}_{ji}) = \sum_p \mathbf{A}_{ji} \mathbf{V}_{ip} \frac{R_{jp}^l}{2 \, \mathbf{O}_{jp} + \epsilon} \tag{A.9}$$

Finally, in order to mitigate noise in the attributions for Vision Transformers, the authors [2] propose to apply the LRP-z^+ on the Deep Taylor linearization of the softmax non-linearity Eq. (A.6):

$$R_i^{l-1} = \sum_j (\mathbf{J}_{ji} \; x_i)^+ \frac{R_j^l}{\sum_k (\mathbf{J}_{jk} \; x_k)^+ + \tilde{b}_j^+} \tag{A.10}$$

where J is the Jacobian of the softmax computed at the current point x and $\tilde{b}$ is the bias term of the linearization. For more details, please refer to [2].

A.5 Faithful LRP Composites

The works of [2, 24] introduced 2 different LRP composites as faithful explainer [37]. For CNNs, [24] suggested applying LRP-z^+ to FCLs and LRP-ϵ to other remaining layers (including HLLs, MLLs, LLLs).

On the other hand, experiments from [2] over the layers of transformers (differentiated based on layer types) demonstrated simple LRP-ϵ (Eq. (A.3)) generates faithful explanation for Large Language Models (LLMs). However, unlike LLMs, the composite of Tab. A.1 seems to be more suitable for ViTs.

B Spatial Dimension of To-Be-Pruned Components

In Eq. (1), $R_p(x_i)$ presented the relevance of component p for reference sample x_i. However, it is crucial to discuss the spatial dimensionality of each component

Table A.1: Faithful LRP composite for Vision Transformers. LinearInputProjection and LinearOutputProjection are the linear layers generating Q, K, V, and O inside attention-module of Eq. (A.7) and Eq. (A.6). The values of γ for applying LRP in Convolution and Linear layers are 0.25 and, 0.05 respectively.

Conv.	Linear	LinearInputProjection	LinearOutputProjection	Softmax
γ	γ	ϵ	ϵ	Attn-LRP(z^+)

depending on its type because the relevance of a component gets aggregated over its dimensions. Convolution filters of CNNs have the size of the tuple (h, w, d) indicating the height, weight, and depth of a filter. Thus, for a batch of size n and convolution filter p, Tab. A.1 will be transformed to:

$$\bar{R}_p = \sum_i^n \sum_j^h \sum_l^w \sum_r^d R_{(p,j,l,r)}(x_i) \tag{A.11}$$

Unlike linear layers of CNNs which simply follow Eq. (1), an extra token axes (t) for neurons inside linear layers of transformers resulting in this modification:

$$\bar{R}_p = \sum_i^n \sum_j^t R_{(p,j)}(x_i) \tag{A.12}$$

Attention heads of Eq. (A.6), have two extra axes (d_q, d_k) regarding as Query, and Key. Aggregation of relevance follows below formula:

$$\bar{R}_p = \sum_i^n \sum_j^{d_q} \sum_l^{d_k} R_{(p,j,l)}(x_i) \tag{A.13}$$

However, in cases of computing the magnitude of relevance, aggregation should follow this structure:

$$\bar{R}_p = \sum_i^n \sum_j^{d_q} \big| \sum_l^{d_k} R_{(p,j,l)}(x_i) \big| \tag{A.14}$$

C   Additionally on Overparameterization

Similar to Fig. 3, Fig. A.1 demonstrates the effect of overparameterization additionally on VGG-16 and ResNet-50 signifying again over-use of parameters in simpler tasks. Similarly in this case, architectures with skip connections show more potential to be overparameterized.
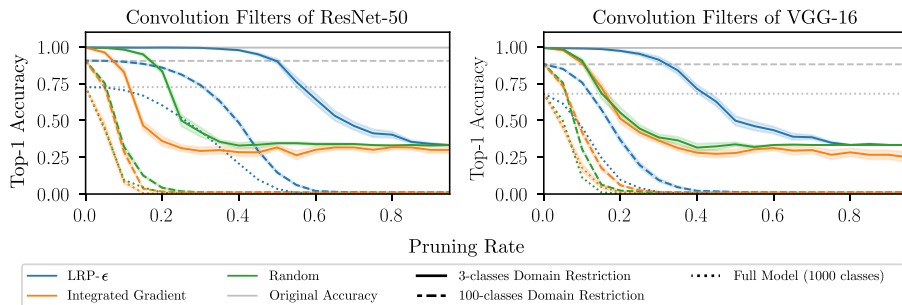
Fig. A.1: With settings similar to Fig. 3, few-shot pruning demonstrates overparameterization exceeds, especially in cases of simpler tasks. Additionally, *Relevance* denoted in the legend is computed by LRP framework with ϵ rule (see Appendix A.2) applied across all layers.

D On Optimization: From Bayesian to Grid Search

To find an optimal LRP composite for pruning, our approach takes place by discovering prospective parameters (Sec. 3.3) via Bayesian Optimization [15] (with Gaussian Process regressor [46] as a surrogate model back-bone), followed by Grid Search on the reduced parameter space. This seems to be a more effective solution rather than naively applying Grid Search over the whole parameters when we can reduce from 50% up to 90% of our search space approximately.

E On Number of Reference Samples

Fig. A.2 indicates and validates the findings of work [48] that using minimally 10 reference samples results in stable pruning for CNNs, as no more impact can be observed when using more samples. The fewer used samples per class lead to a lower pruning rate.

F Optimized Composites

F.1 Top Composites for Pruning

In later tables, based on the conducted optimization, we demonstrate top composites for pruning on CNNs (Appendix F.1), ViT-B-16 (Appendix F.1) and propose a general composite with persuasive performance to use over each architecture type individually. Later, in Appendix F.1 we discuss which parameters are more important to be tuned.

F.2 Evaluating the Composite

As described in Sec. 3.3, we proposed our pruning framework as a means of evaluating attribution methods in terms of aligning with the actual importance
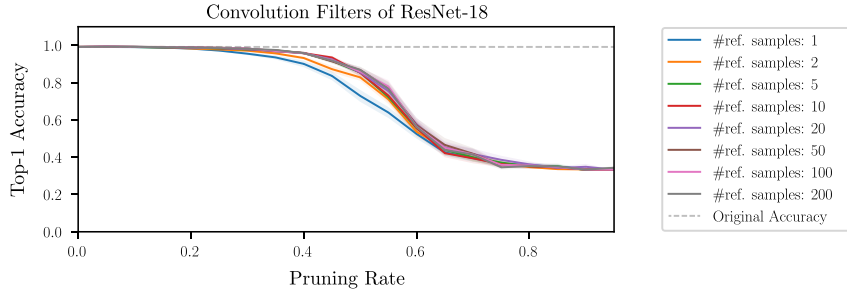
Fig. A.2: Tracking changes of pruning rate for a ResNet-18 pre-trained on ImageNet based on the variable number of reference samples to conduct pruning via LRP for 20 different random seeds. We used LRP-ϵ over all layers.

Table A.2: Top composites per architecture for Pruning Convolution Filters of CNNs. The last column demonstrates the highest compression rate while keeping 95% of baseline accuracy (with $\pm$ 3% of approximation error). Last row indicates using LRP-ϵ (with $\epsilon = 1e - 6$) is a generally strong indicator for pruning these CNNs. Based on section Sec. 3.3, the True flag of Magnitude Flag (MAG) suggests that sorting the importance of components should be based on the magnitude of the relevance rather than its sign.

Models	LLL	MLL	HLL	FC	MAG	$(\uparrow)A^{PR}$	Top-PR (%)
VGG-16	ϵ	ϵ	ϵ	ϵ	True	0.608	26
VGG-16-BN	ϵ	ϵ	ϵ	ϵ	True	0.609	28
ResNet-18	ϵ	ϵ	ϵ	$\alpha^2\beta^1$	True	0.691	41
ResNet-50	ϵ	ϵ	ϵ	ϵ	True	0.718	45
Across All	ϵ	ϵ	ϵ	ϵ	True	0.656	

Table A.3: Top composites for pruning Linear Layers and Attention Heads of ViT-B-16. Similar to the previous table, **Top-PR** demonstrates the highest pruning rate while keeping 95% of baseline accuracy (with $\pm$ 3% of approximation error). The last row also proposes a composite suitable to both targeted structures. $\epsilon = 1e - 6$ and $\gamma = 0.25$ have been assigned to compute LRP-ϵ and LRP-γ

Structure	LLL	MLL	HLL	FC	Softmax	MAG	$(\uparrow)A^{PR}$	Top-PR(%)
Attention Heads	ϵ	ϵ	z^+	$\alpha^2\beta^1$	ϵ	True	0.851	57
Linear Layers	ϵ	$\alpha^2\beta^1$	γ	γ	CP-LRP	True	0.871	66
Across All	ϵ	ϵ	ϵ	z^+	z^+	True	0.851	

of latent components. However, it is noted that composites highly scored by this criterion, are not necessarily performant in generating explanation heatmaps. Our results in Sec. 4.3 exhibited that LRP composite (Appendix A.5) suggested

Table A.4: The importance of tuning an individual parameter that is involved in constructing LRP composite and pruning framework, can be assessed by computing the deviation in performance when that specific parameter is altered while others remain constant. The most influential ones per architecture are displayed in bold font.

Model	LLL	MLL	HLL	FC	Softmax	MAG
VGG-16	0.028	**0.075**	0.047	0.030		0.120
VGG-16-BN	0.009	0.062	**0.072**	0.025		-
ResNet-18	0.019	-	**0.39**	0.17		-
ResNet-50	0.059	**0.72**	0.070	0.017		0.205
ViT-B-16 (Linear Layers)	**0.102**	0.077	0.05	0.012	0.006	-
ViT-B-16 (Attention Heads)	0.012	**0.048**	0.025	0.029	0.012	**0.061**
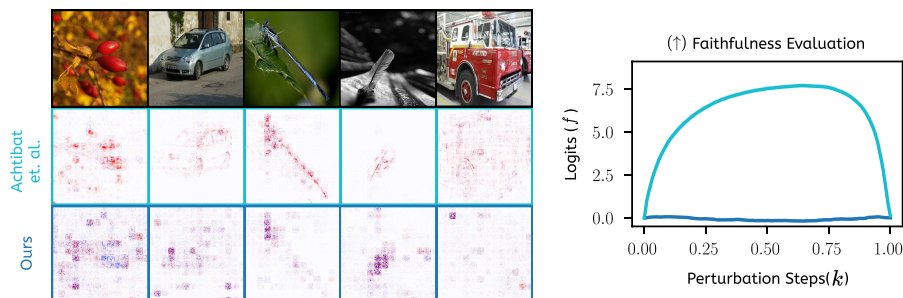
Fig. A.3: Qualitative experiment of ViT-B-16 on the *left* shows that the composite from [2] generates more localized and less noisy explanations in the input-space rather than ours, proposed in Tab. A.1. A quantitative experiment of evaluating attribution methods on *right* side of the figure validates this finding. The higher area under the curve induces a more faithful heatmap. This experiment took place by using around 1000 samples from ImageNet and 100 perturbation steps (see work of [7] for the description of this metric).

by [2] is not the most suitable option for pruning. However, qualitative and quantitative experiment (based on [2, 4, 5]) in Fig. A.3 demonstrates that our best composite (see Tab. A.1) is not proper to render explanation heatmap. All in all, LRP explainers tuned to perform well in pruning, do not necessarily induce reliable heatmaps.

G   Layer-wise Relevance Flow

Later figures (A.4, A.5, A.6, A.7, A.8, A.9) partially elucidate the pruning strategy by illustrating relevance magnitude of components in a layer-wise fashion.
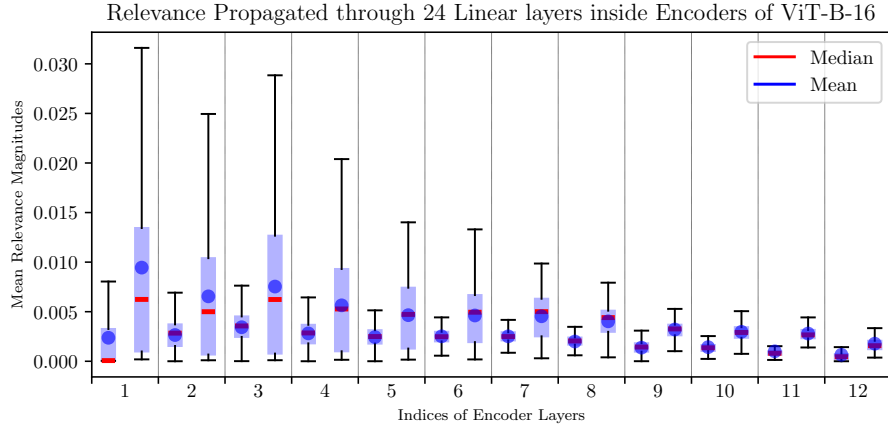
Relevance Propagated through 24 Linear layers inside Encoders of ViT-B-16

Fig. A.4: There are two linear layers inside 12 `Encoder` layers of ViT-B-16. Magnitude of LRP-ϵ relevance shows higher magnitudes as we approach lower layers. Interestingly, the second layers in each `Encoder` block has generally higher relevance values, which can be expected, as here 512 instead of 2048 neurons (as in the previous layer) are included.

H   Change in Behaviour

In later figures (A.10, A.11, A.12), we qualitatively demonstrate examples of pruning based on 3 randomly chosen classes and trace changes in the explanation heatmap of the pruned model.
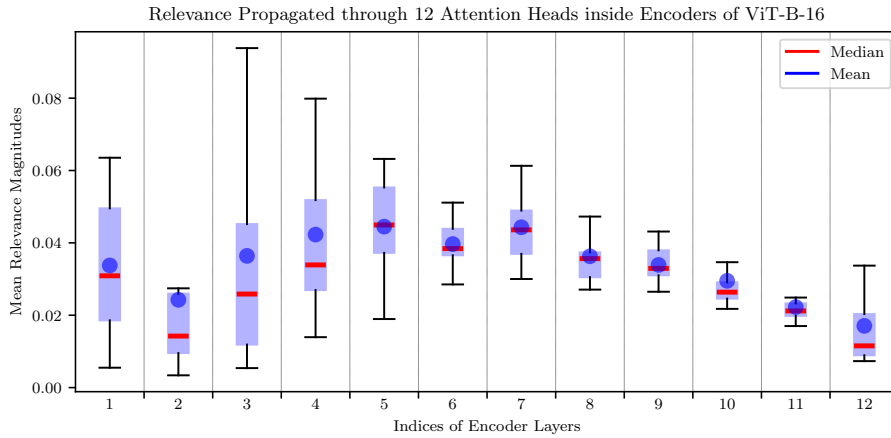
Fig. A.5: 144 attention heads are distributed in 12 `Encoder` layers of ViT-B-16 highlight higher relevance in mid-level layers, corresponding to higher importance in decision-making.
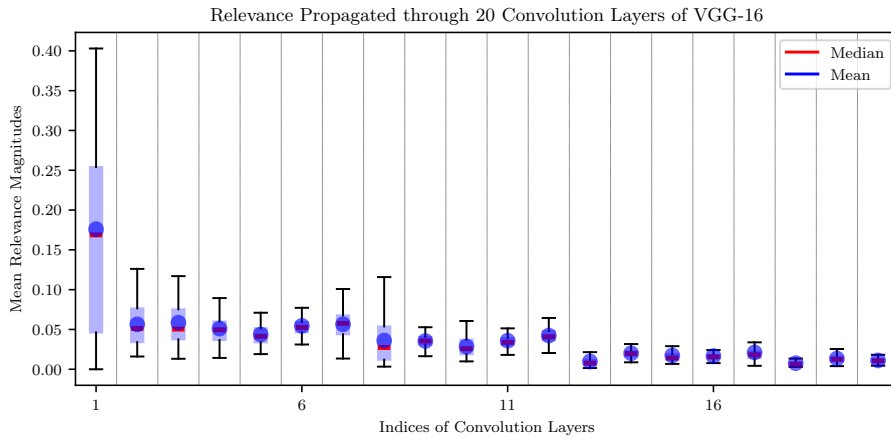
Fig. A.6: Generally higher relevance for mid to low level convolution filters. There are in total, 4800 convolution filters in ResNet-18 distributed in 20 layers.
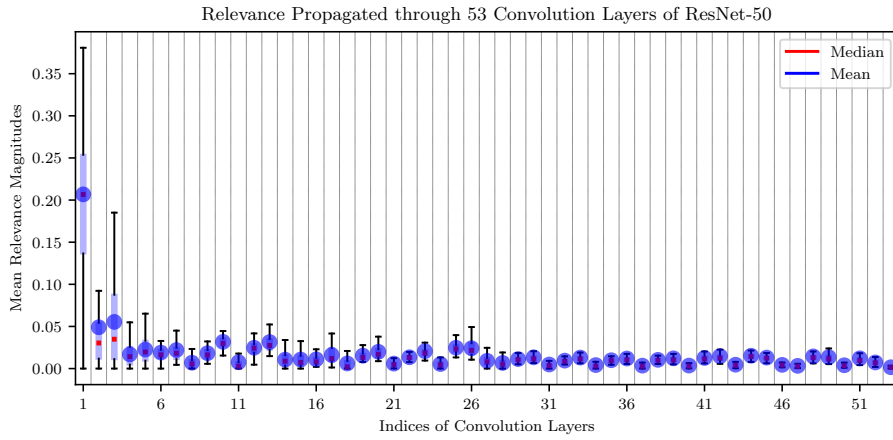
Fig. A.7: There are in total, 26560 convolution filters in ResNet-50 distributed in 53 layers. Higher relevance magnitude in first layers generally stems from having a few number of convolution filters there.
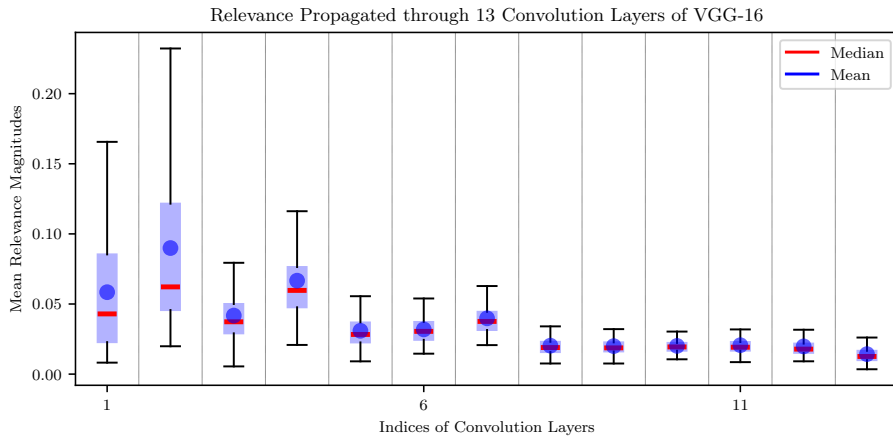
Fig. A.8: There are in total, 4224 convolution filters in VGG-16 distributed in 13 layers. Higher relevance magnitude is visible at mid to lower-level convolution filters.
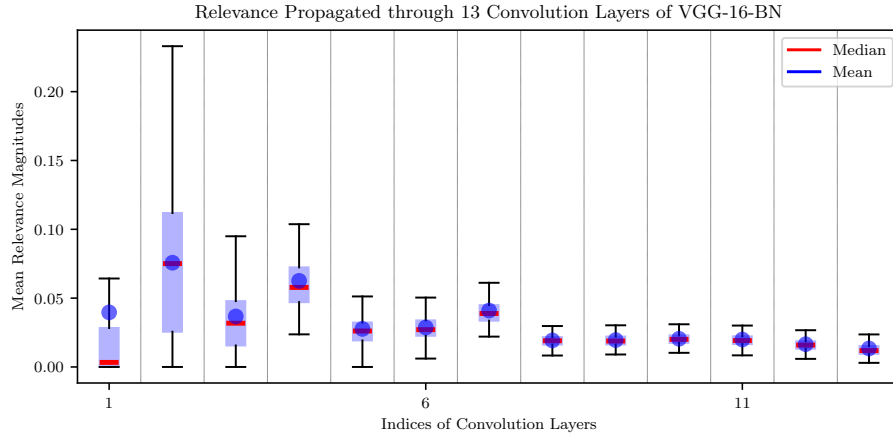
Fig. A.9: Similar to Fig. A.8 there are in total, 4224 convolution filters in VGG-16-BN distributed in 13 layers. Higher relevance magnitude is visible at mid to lower-level convolution filters.
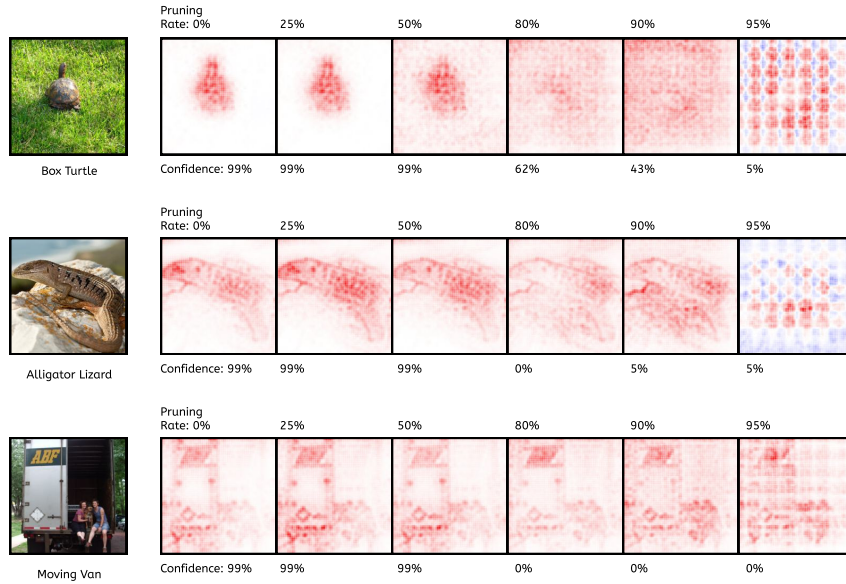

Fig. A.10: Changes in ResNet-18 decision-making for domain restriction of 3 randomly chosen classes (*Box Turtle*, *Alligator Lizard*, and *Moving Van*) via LRP heatmaps (using composite described in Appendix A.5) indicate model failure in high compression rates are caused by high influence of surrounding objects and background despite considering the main object itself. Heatmaps of the model with high pruning rates demonstrate an effect caused by a subsampling shortcut connection becoming apparent.
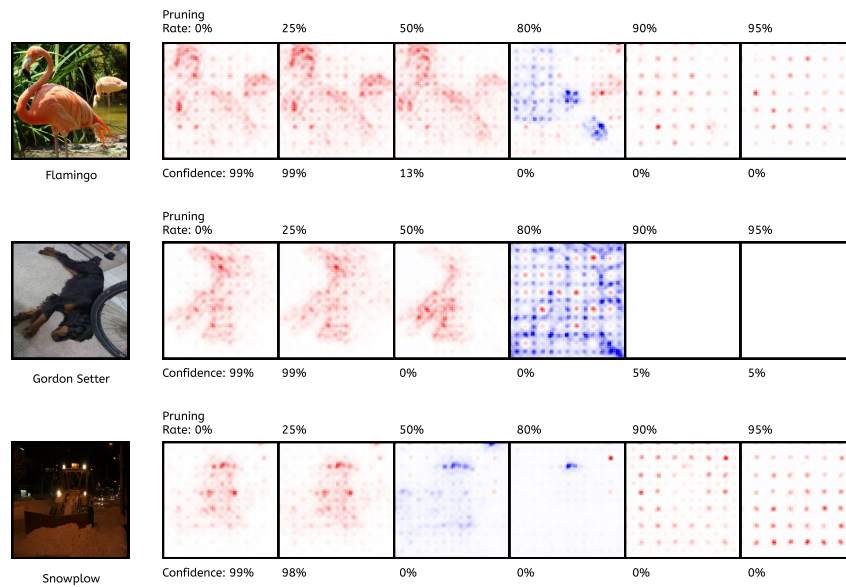
Fig. A.11: Changes in ResNet-50 decision-making for domain restriction of 3 randomly chosen classes (*Flamingo*, *Gordon Setter*, and *Snowplow*) via LRP heatmaps (using composite described in Appendix A.5) indicate model failure in high compression rates are caused by high influence of surrounding objects and unlike ResNet-18 ignores the main object. The shortcut connections of ResNet architecture result in checkerboard pattern exhibited in the heatmaps.
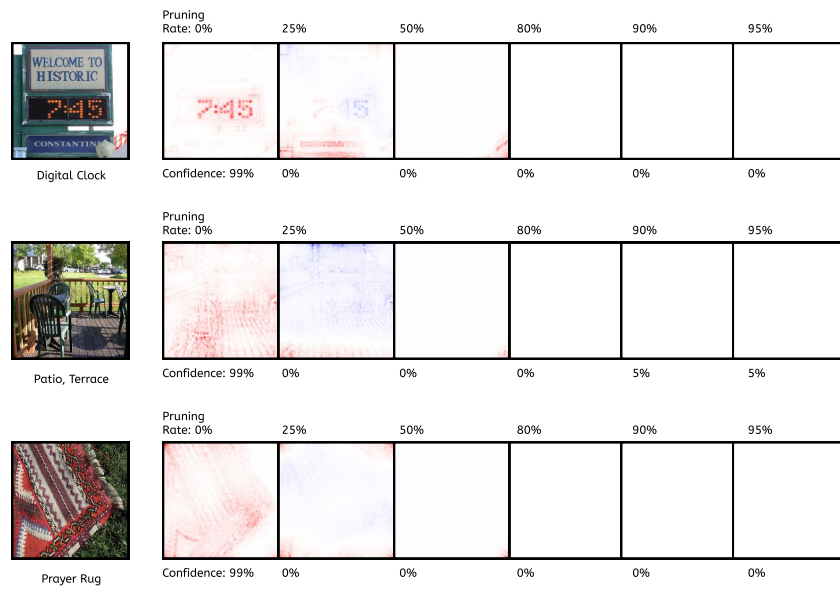
Fig. A.12: Changes in VGG-16-BN decision-making for domain restriction of 3 randomly chosen classes (*Digital Clock*, *Patio, Terrace*, and *Prayer Rug*) via LRP heatmaps (using composite described in Appendix A.5) indicate lower pruning capabilities of this architecture compared to ResNet-18 and ResNet-50.