"Understanding Robustness Lottery": A Geometric Visual Comparative Analysis of Neural Network Pruning Approaches

Zhimin Li, Shusen Liu, Xin Yu, Kailkhura Bhavya, Jie Cao, James Daniel Diffenderfer, Peer-Timo Bremer, *Member, IEEE,* Valerio Pascucci, *Member, IEEE*

Abstract—Deep learning approaches have provided state-of-the-art performance in many applications by relying on large and overparameterized neural networks. However, such networks are very brittle and are difficult to deploy on resource-limited platforms. Model pruning, i.e., reducing the size of the network, is a widely adopted strategy that can lead to a more robust and compact model. Many heuristics exist for model pruning, but our understanding of the pruning process remains limited due to the black-box nature of a neural network model. Empirical studies show that some heuristics improve performance whereas others can make models more brittle. This work aims to shed light on how different pruning methods alter the network's internal feature representation and the corresponding impact on model performance. To facilitate a comprehensive comparison and characterization of the high-dimensional model feature space, we introduce a visual geometric analysis of feature representations. We evaluated a set of critical geometric concepts decomposed from the commonly adopted classification loss and used them to design a visualization system to compare and highlight the impact of pruning on model performance and feature representation. The proposed tool provides an environment for an in-depth comparison of pruning methods and a comprehensive understanding of how the model responds to common data corruption. By leveraging the proposed visualization, machine learning researchers can reveal the similarities between pruning methods and redundancy in robustness evaluation benchmarks, obtain geometric insights about the differences between pruned models that achieve superior robustness performance, and identify samples that are robust or fragile to model pruning and common data corruption.

Index Terms—neural network pruning, robustness, XAI, information visualization

1 Introduction

Recent developments in deep learning have produced significant advances in a variety of application areas [1], [2], [3]. However, such performance is often achieved through extremely large neural networks that consume substantial resources. Further, these models are difficult to deploy and prone to overfitting, leading to poor generalization and fragile behavior [4]. Network pruning, which removes neurons and/or weights from a model, is a common approach to mitigate some of these challenges since compressing models can reduce both their computational footprint and their inherent redundancy [5], [6] without significant performance loss. Although model pruning can be as accurate as the original dense models, some recent works [7], [8] have demonstrated that the resulting sparse models are brittle to out-of-distribution shifts [9]. For example, common, real-world corruptions can reduce the accuracy of such models by up to 40% for images from ImageNet-C [10]. This degradation of robustness has raised serious concerns about the practical viability of pruned models, especially in safety-critical applications such as autonomous driving.

- *Zhimin Li and Valerio Pascucci are with the Scientific Computing and Imaging Institute, University of Utah.E-mail: {zhimin, pascucci}@sci.utah.edu*
- *Xin Yu and Jie Cao are with the School of Computing, University of Utah. E-mail: {xiny, jcao}@cs.utah.edu*
- *Shusen Liu, Kailkhura Bhavya, Diffenderfer James Daniel, and Peer-Timo Bremer are with Lawrence Livermore National Laboratory. E-mail: {liu42, kailkhura1,diffenderfer2, bremer5}@llnl.gov*

Recent results [11] have demonstrated both theoretically and empirically that these problems are a byproduct of the pruning methodologies rather than a fundamental limitation of sparse networks. Researchers have theorized that sparse networks with accuracy and robustness comparable to dense models exist. Furthermore, in some instances, pruning has been empirically demonstrated to improve both the accuracy and robustness of models compared to their dense baselines. This finding is especially surprising as making any model, let alone a pruned version, more robust to out-of-distribution shift has proven difficult. Nevertheless, why certain pruning techniques positively or negatively affect robustness remains unclear. Providing an in-depth understanding will not only support a real-world deployment of such models but also might lead to even more advanced pruning approaches. To date, it is unclear what properties of these models can be attributed to their improved performances, and the model pruning community does not have comprehensive introspection tools to answer these important questions. Such an effort is hampered by the opaque nature of neural networks and the lack of a dedicated system for model comparison and evaluation in the context of neural network pruning.

In this paper, we aim to fill this crucial gap by introducing a visual analytical system for understanding differences among representative pruning methods and measuring and interpreting model behavior under various pruning strategies. As a general goal, we hope to understand the effect of model pruning on multiple levels, e.g., why some sam-

ples are more affected by pruning [7], why certain pruned models [12] can have better generalization performance than a state-of-the-art dense-weight trained model, and how the performance of pruned models differs for unseen or corrupted data, etc. To achieve the goal of building a comprehensive introspection system for analyzing model pruning, our tool focuses on both the computation and visualization fronts.

On the computation side, one essential challenge arises from the need to compare the latent representations of models to understand how making changes to them affects the final prediction. However, a neural network's feature representation usually lies in a high-dimensional space that contains hundreds if not thousands of dimensions without explicit semantics or labels. Comparing such spaces is a nontrivial task, especially considering the behavior of a classifier can be sensitive to small changes (e.g., adversarial example) in the feature representation. Traditional dimensionality reduction methods [13], [14] are not suitable solutions because, for complex latent spaces, they will invariably induce information loss that could significantly impact the trustworthiness of the downstream analysis. A potential solution to this challenge is to preserve high-dimensional relationships in the data for our comparison task. Since our goal is to understand how latent space changes affect the final prediction, e.g., image classification result, what aspects of the feature representation directly contribute to the prediction is critical. As long as this information is encoded faithfully, then the comparison of the high-dimensional feature representations can be more meaningful.

We leverage a set of geometrically inspired features (namely *Angle*, *L2 Norm*, *Margin*) derived from a direct decomposition of the classification loss function (i.e., cross-entropy) to support model pruning comparison and understand the impact of data corruption on model prediction. These geometric features capture aspects of the feature representation that are directly linked to the model's prediction, which allows us to achieve a comparison of network representations by isolating the most crucial information while removing other variations and noises. Moreover, since crucial insights can be obtained only through comparison among different methods, the overall design of the linked interfaces is centered around the ability to provide a contrastive visualization.

Based on the analytical framework, we design a novel visualization system that supports three-level comparisons: pruning methods and evaluation benchmark comparison, model feature representation comparison, and detailed sample comparison. By utilizing the proposed visual analytic system, researchers can compare pruning methods, reveal the similarities of robustness evaluation benchmarks, understand where and how pruning methods differ, identify if a subset of samples is vulnerable to model pruning and data perturbation, and provide insights into why one model is more robust than another. These observations provide feedback to streamline our analysis, improve our understanding of neural network pruning, and motivate useful hypotheses for domain experts to improve a model's performance. Furthermore, we perform a quantitative evaluation that involves ImageNet (see supplementary materials for more image datasets evaluation) to reveal the correlation between geometric features in both data corruption and model pruning.

Our key contributions are summarized as follows:

- A new dedicated introspective visualization system with a three-level hierarchical comparison based on geometric features for analyzing and comparing major pruning methods over different CNN architectures, corruption datasets, and samples.
- A qualitative user study and extensive use cases that involve state-of-the-art models to demonstrate the usability of the proposed visualization system for pruning and robustness analysis.

2 Related Work

In this section, we discuss various research directions that are related to neural network pruning and visualization approaches and discuss their relationship to the proposed approach.

2.1 Network Pruning

In this work, we focus on evaluating and comparing neural network pruning approaches [7], [12], [15], [16], [17], most of which originate in the ML community. LeCun et al. [15] proposed a pruning method based on the assumption that an optimized neural network model can reach a function's minimum, and its second derivative can indicate the importance of weights. Frankle and Carbin [16] proposed a lottery ticket hypothesis that a sparse subnetwork with the same initialization can be as accurate as a dense network after training. Ramanujan et al. [17] and Diffenderfer and Kailkhura [12] showed that an untrained subnetwork has the same performance as a weight-trained model. Hooker et al. [7] introduced pruning-identified exemplars (PIE), which highlight a subset of samples that are more vulnerable to pruning than the other samples. To provide a more comprehensive understanding of these techniques, we discuss the pruning problem in depth and explain the differences among popular pruning methods in Section 3.1. In the visualization community, apart from the aforementioned interactive network pruning work [18], [19], [20], [21], [22], the majority of related works on network pruning have focused on the model interpretation problem.

A notable visualization work in this context is CNNPruner [18]. Li et al. designed a visual analytic system that enables users to interactively perform pruning and explore the trade-off between the model accuracy and pruning ratio. However, their motivation and goal arise from the question of how to design a human-in-the-loop interactive pruning system. Instead, we aim to evaluate and understand different pruning methods and their robustness to a model's internal representation. Particularly, our framework provides a geometric similarity comparison between pruning methods and geometric insights into samples that are more vulnerable/robust to network pruning. Our system finds that the random untrained subnetworks that are surprisingly robust to common corruptions have a significant geometry shape compared with regular well-trained models.

2.2 Data Corruption Evaluation

Neural networks have shown superhuman performance on clean test datasets, but they fall short on robustness by performing poorer on out-of-distribution data [9]. This brittleness issue is more prominent for pruned models [7], which makes it crucial to evaluate them on common corruptions arising in real-world applications. Hendrycks and Dietterich [10] developed corruption robustness benchmarking datasets CIFAR-10/100-C, ImageNet-C, and ImageNet-R to facilitate robustness evaluations of CIFAR and ImageNet classification models. Sun et al. [23] and Mintun et al. [24] further designed new corruption types to complement [10]. In addition to image classification, benchmarking datasets for object detection and point cloud classification were developed in [25] and [26], respectively. In this work, we not only evaluate the model on diverse corrupted datasets but also compare the similarities among evaluation benchmarks and reveal the potential redundancy that helps to streamline the evaluation analysis.

2.3 Visual Model Comparison

Here, we perform a short literature review of different visual model comparison approaches. The common approach compares the input and output of a model to infer its property. Square [27] designed a visualization of a model's output to compare models' multi-label prediction behaviors. Manifold [28] used input and output to compare multiple model behaviors, and different model architectures or algorithms may not constrain the comparison. ConfusionFlow [29] deploys a confusion matrix with a temporal visual encoding that enables users to track class-level temporal information during model training and comparison. StackGenVis [30] utilizes multiple output metrics to compare models' performance and available information to assemble more powerful models. Many techniques [31], [32], [33] use input and output analysis to perform the model comparison. A few surveys [34], [35] also discuss visual models analysis and model comparison techniques.

Information extraction from the output of a model is valuable. It provides a confident score of the model's decisions and reveals the ambiguity of samples among multiple categories. However, this information is limited concerning the stability of the prediction (e.g., adversarial example). Also, recent research [36] has found that output probability can be problematic, and a model may be uncalibrated. Instead of comparing model output, we study the feature representation extracted by a CNN model before classification. By studying the feature representation of a neural network directly, we can gain intuition about the prediction behavior of a model and gain more information about a model's behavior such as the robustness of a prediction. Previous studies have often used dimension reduction that projects high-dimensional data into 2D space to study the data cluster [37] and sample density or outliers. Researches [38] also study how to compare the feature representation between models by preserving global or local information [39], [40]. However, knowledge learned from the 2D projected space contains uncertainty [41] because the projection process may lose a significant amount of information in the original high-dimensional space. How to properly understand and use the projected 2D space from high-dimensional space is still ongoing research.

In convolution neural networks, the last classification layer is linear, and some of these models are attached to a softmax operation. Therefore, the last layer's feature representation is more accessible and interpretable than the previous latent space. Limited work has been designed to understand the latent feature space because of the unknown mysterious structure of the feature latent representation. In this work, we leverage the loss function to construct global geometric features to understand and compare the behavior of multiple pruning methods and different data corruption approaches.

3 Domain Background

In this section, we discuss the basic terminologies, pruning techniques, and evaluation methods used in this study.

3.1 Network Pruning

In this paper, we focus on unstructured pruning, which often removes more redundant network weights than structure pruning. Nevertheless, the analysis pipeline also works for structured pruning, which prunes entire neurons or filters. As summarized in [42], most works in network pruning start with scoring the model parameters based on their potential impact on the network performance, selecting weights of least importance to remove from the network, and optionally performing retraining to gain back performance degradation due to pruning. We will explore the following pruning methods:

Random Pruning: Randomly select a set of weights and remove them from a neural network model.

Magnitude Pruning: Score the weights with their absolute values and prune the ones with the smallest scores [5].

Gradient-Based Pruning: Gradient-based pruning aims to have the least impact on the loss function of the model if removed. Such pruning techniques often utilize the first-order gradient or second-order gradient of the weights (Hessian matrix) [43], [44], [45] to indicate the importance of neural network weights.

Multi-prize lottery tickets (MPTs): This strategy searches for a performant sparse subnetwork within a randomly initialized network and can further compress the network by applying weight binarization. Counter to the traditional training paradigm of learning the network weights, this approach learns which randomly initialized weights should be retained to improve performance by optimizing over surrogate scores that indicate the importance of each weight to network performance. In our experiments, we make use of biprop (Algorithm 1 in [12]). This methodology is built on the *multi-prize lottery ticket hypothesis* [12], which proposes that sufficiently overparameterized randomly initialized networks contain sparse subnetworks that, without any training, can perform comparably to dense networks and are amenable to weight binarization. Theoretical proofs supporting this hypothesis have been established [11], [12], [46].

3.2 Data Corruption Evaluation

Besides accuracy on clean test data, we use the cifar10-C dataset [10], which is the cifar10 test dataset corrupted with 19 common corruption ***algorithms*** from four categories: noise, blur, weather, and digital corruptions. For an image dataset, similar corruption operations can be performed to generate the same corrupted image datasets. For example, the same corruptions have been performed to generate MNIST-C, Cifar100-C, and Imagenet-C datasets. We also perform a quantitative evaluation of these datasets based on observation. These corruptions preserve the semantic content of images, and humans can easily recognize them. Each corruption technique has a severity level from one to five, where a larger number denotes a more severe corruption.

4 Tasks Analysis

As previously mentioned, existing model pruning schemes exhibit diverse behaviors. Specifically, some pruned models are more robust to different data corruption than others. Unfortunately, it is unclear why pruning techniques affect a model in one way and not another. In this work, we aim to provide an in-depth understanding of this phenomenon that will not only support a real-world deployment of such models but could also lead to more advanced pruning approaches.

We target researchers who work on model compression (e.g., pruning, quantization) and model robustness. A fundamental understanding of neural network models and latent feature representation is required. The requirements of this work are based on a long-term regular interview (twice a month over 10 months) with two domain experts who are machine learning researchers. We also collect feedback from the two additional experts not involved in the regular meeting. All four experts are also the coauthors of this work. To help domain experts improve their understanding and answer their questions, we first accumulate these questions and categorize them into three high-level requirements to drive the task design of the visualization system.

R1 - Pruning Method and Evaluation Benchmark Comparison. A comparison of pruning methods is often a task that goes beyond the comparison of two models. Domain experts tell us that they need to compare the behavior of multiple model architectures and evaluation benchmarks over two pruning techniques to tell the differences and similarities between the two pruning methods. This process can be questionable because of the number of models we need to compare, and manual solutions are often hard to scale. Meanwhile, it is also a question of which data corruption benchmarks are more valuable for evaluation. To answer the above question, the process also involves a large amount of model comparisons.

R2 - Model Comparison The model comparison reveals the similarities and differences between the two models. A pruning operation on a network leads to a new neural network model. What is the difference between pruned and unpruned models? Given two models pruned with different pruning techniques, what properties of the feature representation make their robustness differ?

R3 - Samples Comparison Comparing the impact of pruning on a few samples is repeatedly mentioned by the domain experts instead of just observing the model prediction accuracy. Two pruning methods may impact the decision of a sample differently and what feature is dropped by the model can be different. Which part of the samples is more affected by the model and which part is not? What properties make certain samples more vulnerable to model pruning? Will the pruning operation lead to prediction bias?

To address the above requirements, we design five tasks that individually or jointly address them. In summary, T1 and T2 are designed to address pruning method/benchmark comparison (**R1**). T3 is designed to address model comparison (**R2**). T4 and T5 are designed to address sample subset comparison (**R3**).

T1 - Overview Evaluation: Domain experts often perform experiments on multiple architectures, different pruning methods, and various data corruption evaluation benchmarks. A succinct overview of these evaluation results can help domain experts assess the pros and cons of different pruning solutions and architectures and narrow their analysis to the most interesting subset for a detailed examination.

T2 - Compare Pruning Methods and Reveal Evaluation Benchmarks Similarity: Understanding the similarity between pruning and corruption operations helps domain experts quickly highlight the difference between two pruning approaches. The difference will lead to further examination and understanding of what makes the two approaches different from each other. Over a large number of evaluation benchmarks, revealing the redundancy between benchmarks will help optimize the analysis pipeline.

T3 - Compare the Geometry of Feature Representation: Comparing geometry differences between models' feature representations provides domain experts with a visual understanding of how models' feature representations change after pruning. The geometry comparisons contain rich detail and are more informative than input and output comparisons, which can capture the minor difference between the two models.

T4 - Examine Samples' Performance Over Different Pruning and Corruption Configurations: A summary of a model's prediction over different sample subsets is useful for understanding and diagnosing a model's behaviors under different pruning configurations. It helps domain experts understand what mistakes the model will make and what decisions the model will be confident about.

T5 - Reveal Input Saliency for Model Pruning and Data Corruption: Model pruning leads to varying impacts on a model's decisions on different samples. Certain samples are robust to pruning but others are fragile. Understanding what input saliency makes a sample vulnerable or robust is a critical reference for domain experts to reason a model's behavior.

5 Geometric View of Latent Space

Fulfilling the above-mentioned requirements is a nontrivial task. In particular, how to display a model's behavior summary over a large amount of data besides prediction accuracy and how to perform latent space comparisons are difficult questions to answer. To address these challenges, in this section, we discuss the class direction and corresponding three geometric metrics (*angle*, *length*, and *margin*) on a

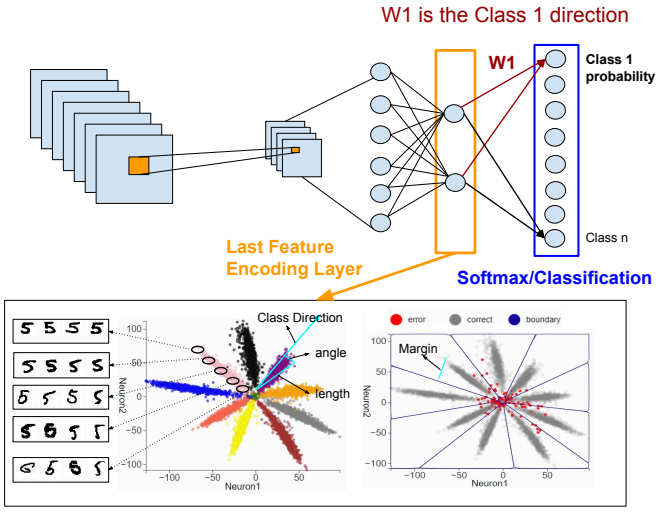

Fig. 1. This study monitors the last feature encoding a layer's latent feature representation to understand the impact of network pruning. The feature representation of data samples in the last feature encoding layer with 2 neurons can be visualized directly. The star shape is the sample's latent space representation of a model trained with the cross entropy loss. The decision boundary in the layer can also be approximated and visualized by sampling the 2D space. Four geometric features that are connected with the loss function can be identified in the visualization.

neural network's last (a network's layer before classification) layer's feature encoding, which is the accumulated result of previous layers' transformations and directly used for a model's final prediction.

As a motivating example, we consider explaining the intuition of what the geometric features look like in the last feature encoding layer. Fig. 1 highlights the critical concepts in the following discussion. The figure visualizes the feature representation of samples in a neural network. This example uses the LeNet5 structure, except the last feature encoding layer is set with only 2 neurons. We use the cross-entropy loss, the data used the MNIST dataset and the final accuracy of the model is 96.2%. Similar experiments can also be performed in other CNN architectures. These 2D latent feature vectors display a star shape, in which samples that are far away from the origin have more distinguishable features, and samples that are close to the origin are ambiguous (refer to support material for more discussion about the geometric shape and loss function). Class direction is the direction across the middle of samples that belong to a certain category. *Angle* and *length* metrics are marked in the plot, which is the angle with the class direction and a sample's distance to the origin, respectively. The plot on the right shows the same feature vectors but with the decision boundaries of the classification. These geometric properties not only exist in 2D space but can also be generalized to high-dimensional space to help summarize crucial aspects of the latent space structure and form the basis for cross-model comparison.

5.1 Class Direction

The loss function used for training the neural network is critical for shaping the geometry of the latent feature embedding. Cross-entropy loss and negative log-likelihood loss are default loss functions used for classification tasks. For a given example x with a ground truth label $y = i$, the loss function can be formulated as $L_{loss} = -log(P(y = i|x))$ where $P(y = i|x)$ is the predicted probability for a model for the label $y = i$ with a value range in $[0, 1]$.

Without loss of generality, we assume that the neural network is composed of two parts: an encoder h that transforms input x into a feature representation vector $\vec{X} = h(x)$ and a classifier c that is used for producing the predicted probability $P(y = i|x) = c(\vec{X})$. We consider that the classifier part c consists of a fully connected layer, and a soft-max activation function, which is a general configuration in convolutional neural network models. $\mathcal{C}$ is the number of classes for a classification task and $\vec{X}$ is the m-dimensional feature embedding of $\vec{X} \in \mathbb{R}^m$. The last fully connected layer in classifier part c parameterized with weight $\boldsymbol{W} \in \mathbb{R}^{m \times \mathcal{C}}$ will take this m-dimensional feature vector as input and project it into $\mathcal{C}$ scores, and then output the predicted probability via the soft-max activation function. These operations are summarized as Equation (1). The predicted probability $P(y = i|x)$ is determined by the dot products of feature representation $\vec{X}$ and each target label j's neuron weight $\vec{W}_j$.

$$P(y = i|x) = \frac{e^{\vec{W}_i \cdot \vec{X}}}{\sum_{j=0}^{\mathcal{C}} e^{\vec{W}_j \cdot \vec{X}}} = \frac{e^{\|\vec{W}_i\| \|\vec{X}\| \cos\theta_i}}{\sum_{j=0}^{\mathcal{C}} e^{\|\vec{W}_j\| \|\vec{X}\| \cos\theta_j}} \quad (1)$$

$$= \frac{1}{\sum_{j \neq i}^{\mathcal{C}} e^{\|\vec{X}\|(C_j \cos\theta_j - C_i \cos\theta_i)} + 1} \quad (2)$$

The dot product operation (denoted as $\cdot$) can be interpreted as the feature embedding $\vec{X}$ of every input example being projected onto each label j's $\vec{W}_j$ direction and multiplied with $\|\vec{W}_j\|$. Here, $\|\vec{W}_j\|$ is a constant, and the predicted probability is determined by the L2 norm $\|\vec{X}\|$ and the angles θ_j between the directions of $\vec{X}$ and each $\vec{W}_j$. Based on the above intuition, we define the fixed weight vector $\vec{W}_j$ parameterizing the classifier part c as the **class direction** of category j in the network's last layer latent space.

5.2 Angle, L2 Norm, and Margin

With the definition of the class direction, we introduce three geometric features of the last layer latent space - angle, length, and margin.

L2 norm metric is the L2 norm $||\vec{X}||$ of feature vectors $\vec{X}$ that affects the confidence or output probability of a prediction.

Angle metric is the geometric angle θ between class directions and feature vectors. A small angle between the class direction $\vec{W}_i$ means the sample will be predicted as label i with high probability. If a feature vector has similar angles with respect to two class directions, then the model will assign similar probability to both categories. If the model makes an incorrect prediction on a sample, then the feature vector of this sample often has a large angle with the target class direction. This metric can be affected by the curse of dimensionality. See support material for more details.

Margin metric is the minimum distance to the decision boundary that is often interpreted as a prediction's robustness. A large margin can tolerate severe data corruption and large perturbations in the input sample. In Equation (1), a sample with the feature embedding $\vec{X}$ belonging to label i needs to have the largest output probability of the model that needs to satisfy

$$\vec{W}_i \cdot \vec{X} - \vec{W}_j \cdot \vec{X} > 0, j \neq i, j \in 1, .., \mathcal{C}.$$

The decision boundary of label i is constructed with $n-1$ hyper-planes $(\vec{W}_i - \vec{W}_j) \cdot \vec{X} = 0, j \neq i, j \in 1, .., \mathcal{C}$. The minimum distance of a feature vector to the decision boundary is the minimum distance of the feature vector X to all $n-1$ hyper-planes:

$$min\{\frac{\|(\vec{W}_i - \vec{W}_j) \cdot \vec{X}\|}{\|(\vec{W}_i - \vec{W}_j)\|}, \quad j \neq i,\ j \in 1, .., \mathcal{C}\}. \tag{3}$$

Note that if a model makes a wrong prediction on a sample, then the margin value will be multiplied by a negative one to indicate the error.

Currently, the proposed geometric features analysis is derived from the loss function and designed for classification tasks. They are independent of model architectures (e.g., CNN and Transformer). Similar concepts are discussed for model optimization [47], [48] for face recognition tasks but limited works discuss model interpretation and comparison. Our work combines these metrics to describe the geometry of a neural network's high-dimensional feature representation. It embeds these metrics into a visualization system to compare network pruning methods and data corruption evaluation.

6 System Design

Leveraging previously proposed geometric metrics, we design a visualization system that generally follows the overview first (Fig. 2(ⓑ) and detail-on-demand (Fig. 2ⓒ, ⓓ) mantra [49].

The process in Fig. 2 (ⓐ) is the classical work pipeline of machine learning engineers for performing model pruning [50], which provides model prediction accuracy only for comparison. Beyond the classical pruning analysis workflow, our system extends the study to a broader understanding [51] with a three-level comparison scheme. Users can start the analysis by having an overview of all models' performance and comparing pruning methods and evaluation benchmarks. A further investigation into the difference in pruning methods needs to dive into comparing the two models. Furthermore, during the process, users can examine a subset of samples' reactions to the network pruning such as performance reduction and input saliency change. In the following section, we will discuss the design rationale for these visual components and how they work together to address the corresponding domain requirements (see Section 4).

6.1 Evaluation Table and Geometric Similarity View

Having an overview of models' performance is an essential first step (**T1**) for analysis. This process involves comparison among multiple model architectures, pruning ratios, and evaluation benchmarks. Previous works [33], [52] have compared models through accuracy, model parameters, and training loss with a limited understanding of the latent feature representation, and how the model used the representation to make the final decision. The previous section explains the geometry of feature representation and decomposes a network prediction into multiple geometric features for decision and robustness understanding. However, efficiently and scalably comparing models' feature representations and performance can still be challenging, and such an operation is necessary for pruning and evaluation comparison.

To address the above challenge, we design an evaluation table view for performance comparison and a geometric similarity view to scalably compare geometric similarity between models' feature representations. In Fig. 3 ①, the evaluation table view displays an overview summary of models' performance over different data corruption evaluations [10]. The x-axis of the evaluation table view represents different data corruption benchmarks, and the y-axis represents a hierarchical structure that starts with model architectures, pruning methods, and then related pruning ratios. In the visualization, each histogram represents the evaluation outcomes (i.e., accuracy) on a given corruption-type dataset, for models with increasing pruned rates, which is defined as the fraction of weights removed by the pruning algorithm. With the increasing pruned rates, the performance of the model often worsens except for the model after retraining.

Comparing network pruning methods (**T2**) is an important task. In our visualization system, we compare pruning methods by measuring their impact on the geometry of different models' feature representations. Two pruning methods having a similar impact on models' feature representation are considered similar, otherwise the opposite. The first step of the comparison is to design a high-dimension vector to describe the overall feature representation. For each feature representation, we use samples' geometric features–*angle*, *margin* and *l2* norm (Fig. 3 ③) to construct the high-dimension vector. Taking 10,000 samples as an example, the related high-dimension vector has 30,000 geometric features. Each histogram in the evaluation table view represents 10 pruned models with a pruning ratio from 0 to 0.9, and each model has 20 corruption evaluations. Each pruned model with a corruption benchmark generates a feature representation. Currently, the evaluation table view (Fig. 3) has 1200 feature representations.

In Fig. 3 ②, the geometric similarity view aims to provide the similarity information about pruned or retrained models' behavior over different data corruption benchmarks. It projects each feature representation's high-dimensional vector into 2D space by the UMAP [53] dimension reduction techniques. In the visualization, each point represents a model's feature representation, and nearby points are more similar than the others. With the assistance of the geometric similarity view, the user can have a similarity overview of all currently available models in the evaluation table view.

The interaction between the evaluation table view and the geometric similarity view can lead to advanced analysis. The evaluation table view displays the performance information, as well as the architecture, evaluation, and

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2024.3514996

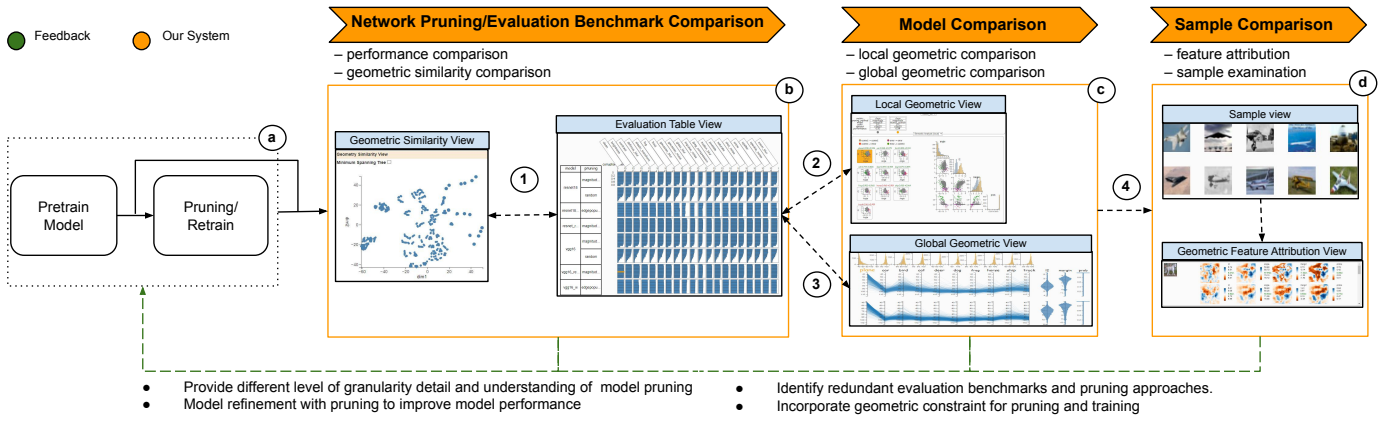

Fig. 2. Compared with classical pruning analysis (ⓐ), the workflow of our visualization system is embedded with three levels of comparison: pruning method and evaluation benchmark comparison; model comparison; and sample comparison. Domain experts can compare pruning techniques and data corruption benchmarks (ⓑ) through the interaction between the evaluation table view and geometric similarity view. For selected models (ⓒ), users can examine and compare their local and global geometric properties. Furthermore, the sample level comparison (ⓓ) enables users to examine features that are captured by models. During the exploration process, users can choose a pruning technique to refine the original model, remove redundant evaluation benchmarks, or incorporate geometric constraints for model retraining.

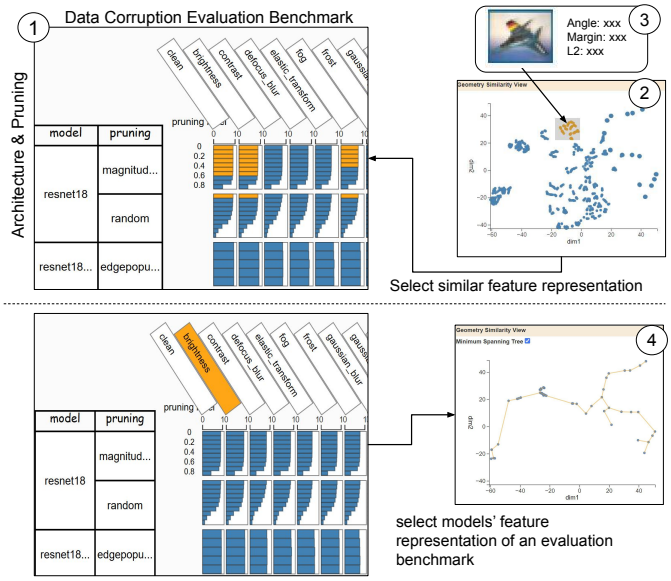

Fig. 3. The evaluation table view and geometric similarity view provide an overview of model performance and feature representation similarity. Users can combine two views to compare the similarity of pruning methods and corruption evaluations.

pruning approach used, while the geometric similarity view provides the similarity information. A set of models, which have similar feature representation, are selected in the geometric similarity view (Fig. 3 ②), and related models and benchmarks are highlighted in the evaluation table view (Fig. 3 ①). These highlight columns belong to Gaussian noise, shot noise, and speckle noise, which indicates that these corruption operations lead to a similar impact on model behavior. Similarly, the model selection operation can also be performed in the evaluation table view, and relative models are highlighted in the geometric similarity view. In Fig. 3 ④, we select models that are evaluated with the *impulse noise* corruption evaluation. The geometric similarity view displays representations that belong only to this evaluation.

During the comparison, each pruning method and evaluation benchmark is related to many models(or points) in the geometric similarity view. Because of visual overlap and a large number of reference points, directly comparing two sets of points in the geometric similarity view to reveal the similarity of the pruning approach is visually difficult. To mitigate this problem, the geometric similarity view allows points that belong to the same pruning methods or valuation benchmarks to be connected by a minimum spanning tree for similarity comparison (see use case 7.1 for more detail). Such a design makes the two-point set comparison easy for domain experts.

6.2 Local Geometry View

The model evaluation table view and geometric similarity view summarize the overall performance and compare the similarity of network pruning methods. The follow-up step is to understand why one pruning method is geometrically different from the other. Therefore, users need to have a detailed geometric examination of each model's feature representation.

The local geometry view performs class-specific evaluations, which helps in conveying how well samples from a specific class are classified and uncovers the cause of potentially poor performance. To demonstrate the usability, the visualization in Fig. 4 ① displays geometric metrics angle and l2 norm for each class to visualize the high-dimensional star shape (Fig 1) of the feature representation.

The accuracy on the top shows that samples from the cat class have 79.4% accuracy, which is less than the rest of the samples, and the car class has the best performance with 96.8% accuracy. The *angle* and *length* metrics are presented as a scatter plot. Within a single model, these geometric metric values often have similar ranges for each class. Among these points, the gray color indicates the correctly classified samples, and the color red indicates the incorrectly classified samples. In all categories, these incorrect samples have larger *angle* and smaller *length* metrics than the correct samples. Here, we select the lowest-performance cat class for a more detailed examination. By selecting a category, the

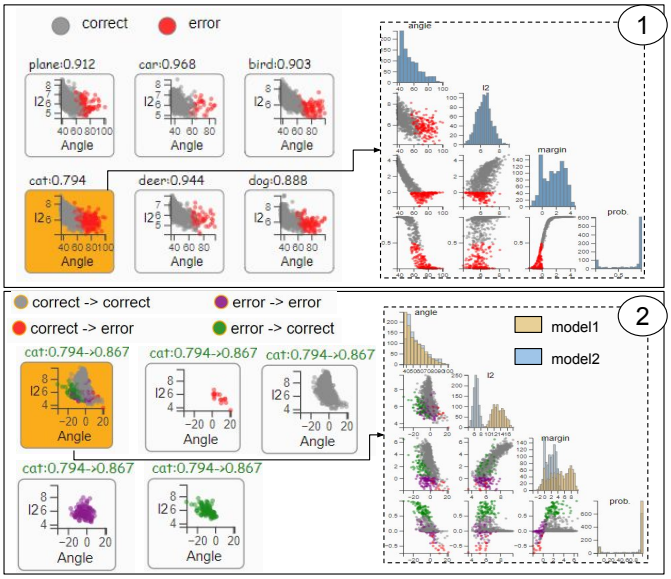

Fig. 4. Local geometric view visually presents the geometric metrics distribution of each label category ①. The incorrect predictions are samples often with large *angle* and small *l2* norm. Furthermore, the local geometric view also enables users to compare two models' geometric differences (②) by displaying the geometric Δ value between samples from different models and providing highlight operations for samples that are errors in one model but correct in the other.

correlated all local features including *angle*, *l2*, *margin*, and *probability* displayed as the scatter plot matrix for detailed understanding.

In addition to exploring a model's latent space geometry and its per-class performance, the local analysis also provides a comparison between models' feature representations (**T3**). In Fig. 4 ②, we select another model with pruning and retraining for comparison. We choose the cat category as an example because the performance is improved by 7% after the pruning and retraining operation. The visualization shows the variation of the geometric metric distribution of cat samples and the corresponding geometric value change. Users can selectively pick samples that exhibit diverse reactions to network pruning and retraining. We encode these reactions into four types based on domain experts' recommendations. One of these types is that a sample is predicted correctly in one model but wrong in the other (red). In the opposite case, a sample is predicted wrong in the original model but incorrect in the new model, which is encoded as green. The value in each axis is the difference (e.g., Δ angle = angle - angle') of the samples' geometric value between the two models. The related scatter plot matrix encodes all geometric feature distributions and their distribution shift. Similarly, the user can select a category (e.g., cat class) to further examine these geometric variations in the scatter plot matrix view.

6.3 Global Geometry View

The local geometry view provides a detailed view of a model class by class. However, for the comparison task, the ability to have a more comprehensive global summary is critical. A global geometry view gives a geometric overview of how a model performs on all classes of the currently selected dataset (**T3**), which shows not only how well samples are classified, but also what other classes the model may confuse with.

The global geometry view uses a parallel coordinate display in a $n+3$ dimension configuration, in which n is the number of classes (for a dataset with a large number of classes, a pres-election can be applied to focus on a subset of classes to make visual encoding and exploration manageable). The first n dimensions represent the *angle* metric of a sample for each class direction. As the previous section described (Section 5), a relatively small angle with a label indicates a good chance the sample belongs to this category.

For example, Fig. 5 ② displays samples belonging to the plane class. Most of the samples have smaller angles in the direction of the plane class in comparison to other classes. However, these plane samples also have relatively small angles of the bird class (Fig. 5 ④) and the ship class (Fig. 5 ⑤), which can lead to incorrect classification. By further examining these samples, we can see a similar shared background (e.g., ocean), which might be the contributing factor to this mistake. The other features (l2 norm, margin, and probability) are displayed as density plots that are separated from angle features because of the difference in semantic meaning and value scale.

Similarly to the local geometry view, the global geometry view also enables the geometric comparison among models and datasets (**T3**). In Fig. 5 ③, the visualization displays the same plane class but with samples that are corrupted with noise. Before corruption, only a few plane samples had a small angle of the ship and the bird classes. However, with the presence of fog corruption, the number of samples that have small angles of both plane and bird increases. The density plot (Fig. 5 ⑦) on the top of each class axis represents the density of the angle value, which is the missing information in the parallel coordinate view. The y-axis represents the density, and the x-axis represents the angle range of a specific class. The gray color highlights the reference dataset (original), and the yellow color highlights the compared dataset (corrupted). In the visualization, the bird class samples' *angle* metrics shift to the smaller values in the presence of corruption. Moreover, the corrupted dataset as a whole has a smaller *margin* and *length* compared to the clean datasets. Such a result indicates that the uncertainty of the model's prediction between plane samples and bird samples increases. Our comparative interface allows us to extensively reason about similarities and differences among models pruned with various techniques. A detailed case study is presented in Section 7.2.

A confusion matrix [29] aggregates many samples' prediction results to highlight the ambiguities between label predictions. However, if a sample is predicted correctly, the information from the confusion matrix will not help to tell whether the model may confuse this sample with other labels or not. The output probability of a model is supposed to reveal the confusing information of a sample, but many studies have found that a model's output probability is often overcalibrated [36], [54], and researchers should be cautious about how to read these probability values. Compared with previous visual encoding, our design visualization and geometric metrics give more detailed information about a model's prediction. In Fig. 5 ⑥, we demonstrate a set

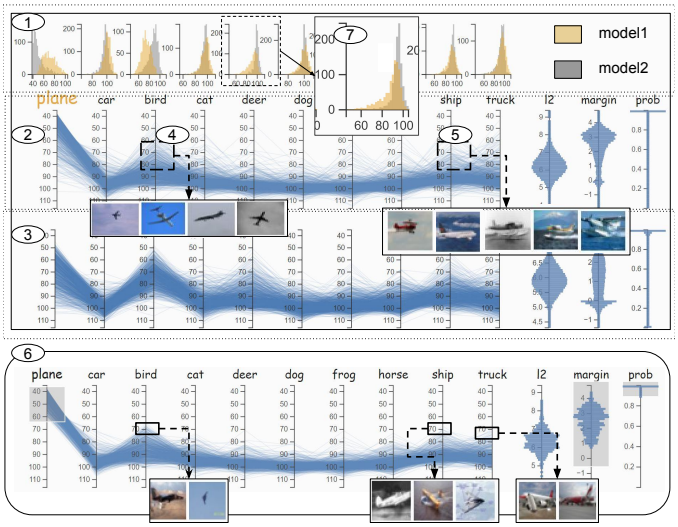

Fig. 5. VGG16 model's behavior over cifar10 plane samples, and the same data but corrupted with fog corruption. In the reference dataset ②, these samples are confused with bird and ship samples. In the fog-corrupted version ③, the confusion is strengthened. In ⑥, the visualization highlights a set of samples that are predicted correctly with high probability but show certain feature ambiguities with the other labels.

of plane samples that are predicted correctly with high probability but have a related small angle with other class labels. These samples are often fragile to data perturbation and pruning, but confusion analysis and output probability do not provide explanations for these vulnerabilities.

6.4 Geometric Attribution View and Sample View

The saliency map [55], [56], [57] of an image tells what kind of features are used by the neural network model for prediction. Beyond the model-level comparison, domain experts are also interested in understanding how the saliency map changes with different pruning methods. The geometric feature attribution view shows how each pixel of an input image contributes to different geometric features. Our method is designed based on a perturbation technique that occludes a part of the input image and checks how much a model changes a sample's latent geometric features compared with the original image (**T5**). The variation of these geometric feature values is visualized as a heat map to highlight input pixels that change.

In Fig. 6, we demonstrate the geometric feature attribution of an image over three geometric metrics and probability from a well-trained VGG16 with different pruning ratios. For example, the heat map of the margin highlights

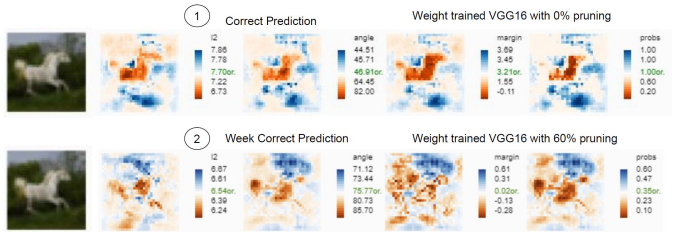

Fig. 6. A geometric feature attribution view gives information about the sensitivity of input pixels of an image. From left to right are the sensitivity heat maps for l2, angle, margin, and probability. From top to bottom are the results of the VGG16 model with a different pruning ratio. Models with different pruning ratios will change the importance of an input image's pixel for the final decision.

the input pixels that are critical for prediction robustness. The color map of each heat map is annotated with a metric value scale. Blue represents the increase in the geometric metric, and red indicates a decrease in the value. Removing the blue region will strengthen a certain geometric feature, and the red region will weaken a geometric feature.

The well-trained VGG16 in Fig. 6 ① predicts the image belonging to the horse category with high confidence. Meanwhile, all four feature sensitivity views highlight the horse's body. This result indicates that for this image, prediction confidence, margin, angle, and l2 norm features focus on similar input. With 60% weights pruned, the model still predicts the image as a horse correctly but with much lower confidence. Compared with the original prediction, the new prediction's four feature sensitivity maps highlight the inconsistent part of the image but with certain portions focused on the horse's body. The sample view (Fig. 2 ⓓ) displays the detailed image of the selected sample during the exploration process.

6.5 Interaction Between Views

In the system, six views coordinate with each other to address domain requirements. We summarize their relationship with each domain task in Table 1, and show how they interact with each other to expand the exploration ability of the visualization system in Fig. 2. Users can interact between the geometric similarity view and evaluation table view to study and compare the similarity between evaluation benchmarks and pruning methods (Fig. 2 ①) and select interesting models for detailed examination and comparison (Fig. 2 ②).

During the pruning analysis, domain experts are also interested in the performance of different architectures and pruning techniques on a subset of samples or a specific class (Fig. 2 ③). To achieve this requirement, during the downstream analysis, our system enables users to select a subset of samples, and the evaluation table view can automatically reflect the performance of the current samples set (**T4**). The current visualization also enables a comparative analysis, which shows the performance comparison between currently selected samples and the entire test dataset.

In either local or global geometric views, users can select a subset of samples and examine their performance on different pruning methods and network architectures in the evaluation table view. Because of the large amount of information in the evaluation table view, we highlight the performance change with different sample subsets with color as Fig. 9 shows. For easy comparison, if the subset has the same performance as the whole dataset, then the histogram bar is blue. If the performance increases on the selected subset of samples compared to the accuracy of the overall test dataset, it is displayed as green. The red shows the opposite revealing a decline in performance. Such an operation enables users to examine more details of model behavior and helps them understand model performance under different levels of granularity. In the end, users can choose a set of samples for detailed image check or a specific sample for feature attribution comparison (Fig. 2 ④).

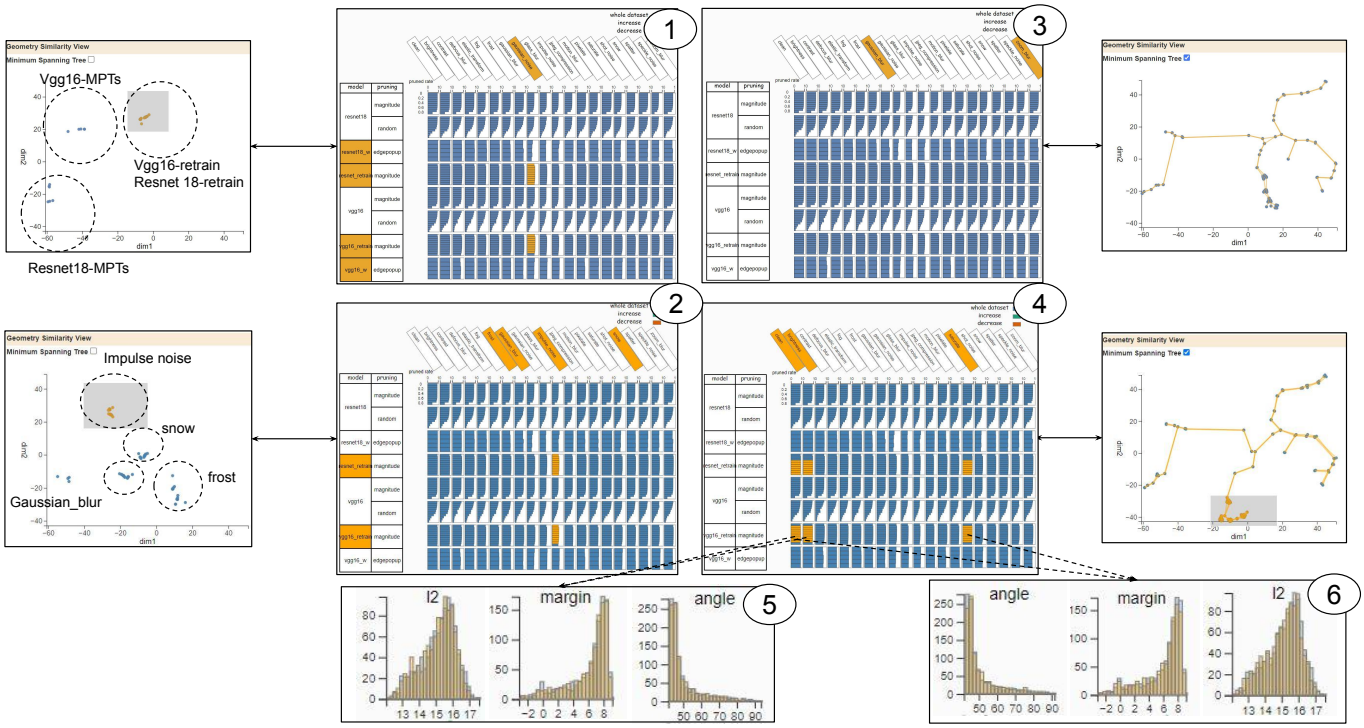

Fig. 7. In the visualization, ① reveal that MPTs pruning approach and magnitude pruning approach end up with different geometric feature representations over Gaussian noise corruption.The finding of ② reveals that pruning and retraining with VGG16 and Resnet18 models have highly comparable geometric feature representations. ③ and ④ together emphasize the existence of redundancy in common corruption evaluation benchmarks. ③ shows redundancy between the Gaussian blur and the zoom blur benchmarks. ④ highlights the similarity among clean, *brightness*, and *saturate* benchmarks. In ⑤, ⑥, we select two pairs of evaluation benchmarks on the VGG16 retrain model for comparison. Their geometric features' distribution are similar and overlap with each other.

TABLE 1
Composition of visual components to address each domain task.

Visual Components\Tasks	T1	T2	T3	T4	T5
Evaluation Table View	✓	✓		✓	
Geometric Similarity View	✓	✓	✓		
Local Geometry View			✓	✓	
Global Geometry View			✓	✓	
Geometric Attribution View					✓
Sample View					✓

7 Evaluation

To evaluate the usability of our system, we perform three use cases and domain expert interviews. The three use cases generally follow the design workflow (Fig. 2). We first demonstrate how to use visualization to compare the similarity between pruning methods/evaluation benchmarks (Section 7.1). Then, we show the detailed model comparison to reveal why one pruning method may be different from the other (Section 7.2). Finally, we narrow down the analysis to the sample level and compare how pruning will impact samples' performance, and how pruning changes samples' input saliency (Section 7.3).

7.1 Network Pruning Approaches and Evaluation Benchmarks Comparison

Exploring and testing the performance of network pruning approaches' over multiple corruption evaluation benchmarks are often the initial tasks for domain experts to understand their differences. This process involves questions of whether different pruning approaches will yield similar or different behaviors. The answer to this question helps researchers understand the pros and cons of different pruning methods. Domain experts tell us that they often use similarity metric centered kernel alignment (CKA) [58] to compare two pruned models' feature representations one at a time. However, as the number of feature representations for comparison increases, this approach becomes increasingly intractable. Therefore, there is a growing need for a scalable methodology that can efficiently assess the similarity among multiple models simultaneously.

In Fig. 7, we illustrate how domain experts employ a combination of the geometric similarity view and evaluation table view to address the above concerns. In Fig. 7 ①, we select the data from the Gaussian noise corruption benchmark to generate the feature representations of different models. Specifically, we apply magnitude and MPTs pruning to VGG16 and Resnet18 network architectures for comparative analysis. From the visualization, we can tell that magnitude pruning, when applied with retraining, on both VGG16 and Resnet18 results in feature representations that have a similar geometry as their dense counterparts. Conversely, MPT pruning over these two network architectures makes significant geometric changes. One intriguing observation of ① is that retrained VGG16 and Resnet18 have comparable feature representation over the Gaussian noise corruption benchmark. However, MPT pruning applied to VGG16 and Resnet18 led to significantly different feature representations. To further validate the consistency of similarity be-

tween retrain VGG16 and Resnet18, we narrow our focus to these two models and evaluate them with additional corruptions such as snow, Gaussian blur, frost, and impulse noise. In Fig. 7 ②, the resulting visualization reveals that the similarities between these two models are consistent across multiple evaluation benchmarks.

During the comparison and evaluation, understanding the coverage of the evaluation benchmark is important to assess the quality of the analysis. In the evaluation table view, there are 20 distinct evaluation benchmarks, and each of them is created by different corruption algorithms [10]. During the design of these corruption algorithms, whether these evaluations will lead to similar model behavior has not been thoroughly explored. Corruption operations that result in a similar model behavior may be redundant and unnecessary. In Fig. 7 ③,④, we show how such similarity can be revealed by our system. ③ highlights two corruption benchmarks, Gaussian blur, and zoom blur, in the evaluation table view and geometric similarity view. Notably, the minimum spanning tree constructed by these models evaluated with the selected benchmarks exhibits a significant overlap. Similarly, in ④, the minimum spanning trees of the models that are evaluated with the corruption evaluation over brightness, saturation, and the clean dataset also overlap substantially. We can further use the local geometric view to compare their local geometric feature variation. ⑤ compares the geometric features distribution of retrain VGG16 over clean and the brightness corruption data, and ⑥ compares the clean and the saturate corruption data. In both cases, the difference between their geometric distribution is subtle. In ⑤ and ⑥, each histogram has two color distributions: orange and steel blue. Because these two histograms are highly overlapped, only one is color histogram displayed at the end. These results suggest that for the current image dataset and tasks, models subjected to the currently available pruning methods have similar reactions to the evaluation benchmark. Consequently, there is an opportunity for optimization, such as removing similar benchmarks, to streamline the analysis process, reduce computation complexity, and minimize cost.

7.2 What Is the Representation Difference Between the Robust MPTs Pruned Models and Others (Dense and Retrained)?

The use case in Section 7.1 gives hints about the reaction of different pruned models over multiple evaluation benchmarks. However, it does not give details about the difference between retrained models and MPTs pruned models. The MPTs method has been proven [12] to produce compact models, which not only have high prediction accuracy and small model size but also models that are significantly more robust to various data corruptions than regular weight-trained models[1]. However, it is still unclear why and how the MPT approach achieves such a performance gain. Here, we show that our visualization tool can help develop hypotheses about answering this question by comparing the latent spaces' geometric structures of the traditional weight-trained VGG16, the retrained VGG16 model with a certain pruning ratio, and the VGG16 model generated by MPT.

1. https://robustbench.github.io/div_cifar10_corruptions_heading

From Fig. 8 ⓐ, the visualization result reveals the significant geometric disparity of the model's feature representation. The related models are numbered for reference. To examine for more geometric detail, we use the global geometric view to understand the difference between models. The model number (1)-(6) in ⓐ is used to refer to models (1)-(6) in ⓑ, ⓒ, ⓓ

In Fig. 8, the panel (ⓑ) shows the geometric difference of samples from the truck class with and without the JPEG corruption in the original weight unpruned trained VGG16 model. The model is trained on the cifar10 dataset with 200 epochs, and the final accuracy on clean test data is 91%. With the same test dataset but corrupted with the JPEG compression, the accuracy of the model drops to 72%. The models can distinguish truck images from samples of other classes even though some samples may be slightly confused with car samples (1). Once the dataset is corrupted, the model displays confusion with multiple categories such as plane/ship (2), and the models' global geometric features on corrupted data are not as coherent as the clean dataset. The samples belong to the truck class, and the prediction accuracy dropped from 93.3% to 82.9%.

The panel (ⓓ) illustrates the same comparison, but using the models generated by the MPTs method, i.e., starting with an untrained VGG16 model and then 90% of the weights are pruned, resulting in a model with 91% accuracy. The performance of the model declines to 83% when tested on the corruption dataset, which is significantly better than the performance of the dense-weight trained model (71.84%). For samples belonging to the truck class, the performance of the model drops only slightly from 94.3% to 91.7%. The *angle* distribution of samples with each class direction displays a distinct pattern, i.e., the dense VGG16 has a much larger variance but smaller mean *angle*, whereas the MPT model has much a smaller variance but larger mean. For MPTs, the global geometric visualization comparison (5), (6) shows that the difference between corrupted and clean data is minor.

A simple potential explanation of such an observation is that the pruned model contains fewer parameters, which may lead to a latent space that contains much less information, and the model's prediction should be less sensitive to the input noise, which leads to a more robust model. To verify this hypothesis, we add an additional comparison (ⓒ (3), (4)), in which the model is pruned with 90% weight but retrained 50 epochs to gain the original prediction accuracy around 91.14%. For the truck class, the prediction accuracy increases to 94.4%, and the accuracy of the corrupted data increases from 82.9% to 85.0%. The new comparison gives positive feedback about the hypothesis that decreasing the number of parameters in a model can improve a model's resiliency to data corruption.

However, the model generated by the MPTs model's prediction accuracy on corruption is still significantly better than the retrained model. This behavior indicates that besides the number of parameters in the model-generated MPTs, the unique geometric latent space of the MPTs model can play a significant role.

After presenting this use case to the domain experts, we collected and summarized their feedback about how this information is helpful to their research as follows: "Such an

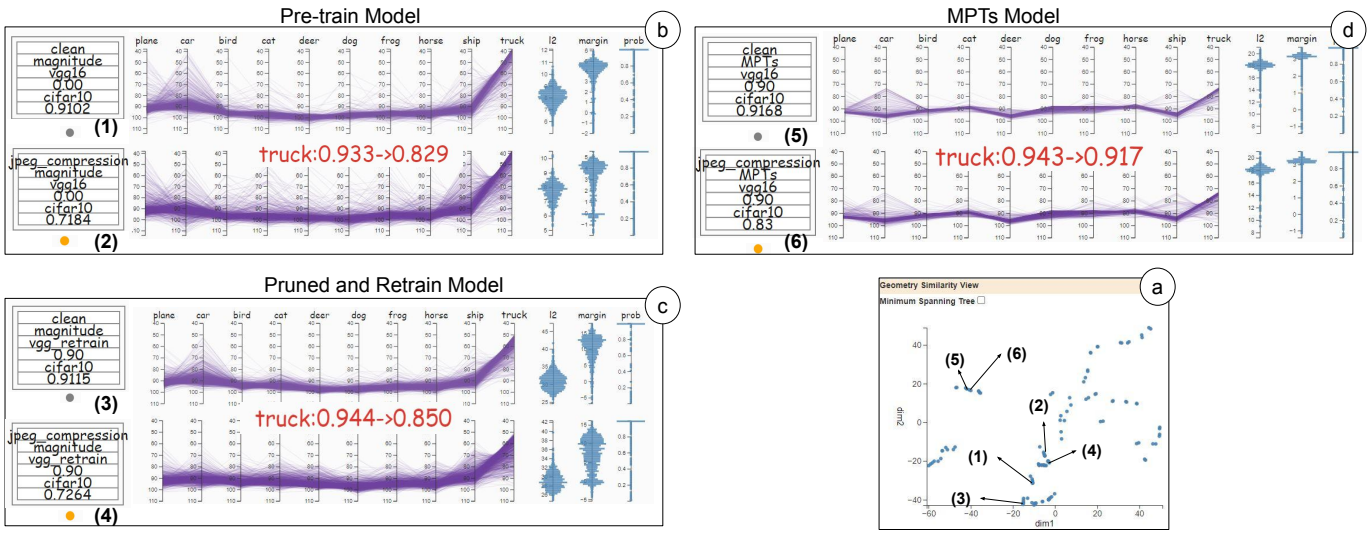

Fig. 8. The geometric similarity view ⓐ highlights the position of related models. The visualization result reveals geometric disparities in their feature representation. The panel ⓒ presents a comparison between truck samples of cifar10 on a 90% weight-pruned retrained VGG16, and the same samples but corrupted with the JPEG compression. The panel ⓑ showcases a dense-weight well-trained VGG16 and its representation of clean and corrupted data. The same comparison ⓓ is performed on the VGG16, which is generated by the MPT method. The JPEG corruption induces pronouncedly more angular variations with multiple class directions and minimum distant shift in the weight-trained VGG16 and retrained VGG16 than the VGG16 generated by the MPTs. This observation provides valuable insights into why the models generated by the MPTs' method tend to be more robust to data corruption compared to the regular weight-trained model and weight-pruned retrained model.

observation motivates a hypothesis that the representation geometry of MPTs is less sensitive to different corruption types than that of the regular weight-trained or retrained models. These sensitive properties may related to the thin standard deviation of geometric features and angle scales of different models' feature representations. Meanwhile, this finding has potentially significant implications for robust machine learning as it suggests that to design a robust neural network, one should not only optimize for the training accuracy but also incorporate additional geometric constraints during training".

7.3 How Are Samples' Geometry Impacted by Model Pruning, Data Corruptions, and Model Retrain?

Model evaluation often aggregates a model's prediction over all test samples, but often misses critical details about the model's behavior in a specific category or a subset of samples. The previous network pruning literature [7] has assessed the impact of network pruning over samples by proposing a metric called PIE (pruning-identified exemplars) to identify vulnerable samples. The PIE value is calculated by the disagreement between compressed and uncompressed models that require a large amount of computation resources. At the same time, this metric does not reveal what properties cause a sample's vulnerability. Here, we demonstrate how we can use our visualization system to quickly highlight these samples and provide depth analysis, such as how pruning and retraining affect a sample and what features are captured or forgotten by the model.

In Fig. 9, we demonstrate an interactive exploration case that compares three subsets of samples' observations with different geometric properties. We select these samples from the frog category of the cifar10 dataset and evaluate their performance on pruned models over available evaluation benchmarks. Fig. 9① displays the evaluation result of samples that are selected from the frog category with large *length* and small *angle*. In the evaluation table view, we can tell that these samples perform better than the whole dataset with high prediction accuracy with different data corruption benchmarks. Meanwhile, they are less affected by the different pruning techniques.

Compared with Fig. 9①, Fig. 9② is the evaluation result from samples that have a relatively larger angle and smaller l2 norms. The overall performance of these samples declines compared with samples from Fig. 9①. These samples demonstrate less resiliency to different pruning and corruption operations. The last case is Fig. 9③, which shows samples with small l2 and a large angle. All these samples have poor performance in different evaluations and different model architectures.

The above observation reveals that large l2 and small angle values often show resiliency to the different pruning operations across multiple network models and different data corruption evaluations. On the other hand, samples with small *length* and large *angle* values display fragile behavior, and these samples are affected dramatically by pruning and data corruption. To further verify such observations, we follow domain experts' suggestions to perform a quantitative evaluation to measure the relationship between geometric properties and data corruption.

Our experiment measures two relationships: how does a sample's geometric properties correlate with different data corruptions, and how does a sample's geometric properties correlate with standard model pruning? In Table 2 and Table 3, we demonstrate our evaluation result over the Imagenet dataset. The result finds that the angle and margin metric are strongly correlated with the samples' resiliency to data corruption and model pruning. However, the l2 norm does not show a consistent correlation with data corruption and pruning. The support material includes details of how we performed the experiment and more dataset evaluation results. These results show that geometric properties can be

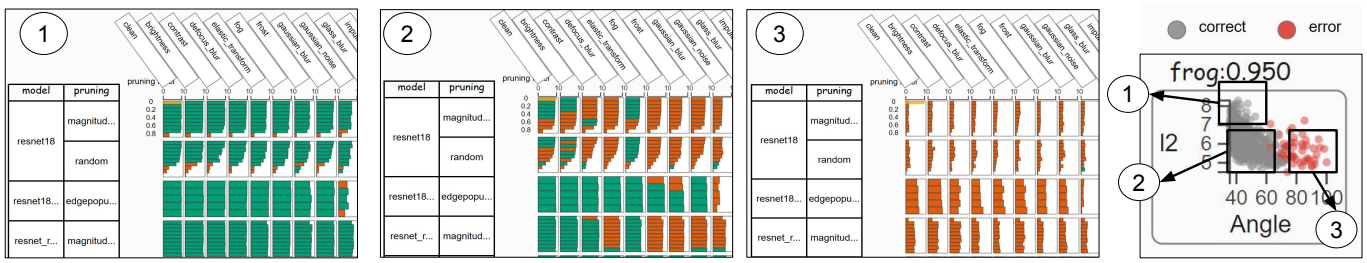

Fig. 9. An overview of the evaluation comparison across multiple architectures and pruning techniques of frog samples. The samples with large *length* and small *angle* are less affected by the pruning, different model architecture, and data corruption. On the other hand, the samples with smaller *length* and large *angle* are fragile over different pruning approaches and models.

TABLE 2
The result of the **Imagenet-C** validation dataset. This table shows the Pearson correlation coefficient between different geometric features and models' robustness (*supplementary equation (1)*). Angle and margin show a significant correlation with robustness. The correlation between l2 and robustness is moderate or subtle.

CNN Architecture	angle	l2	margin.
VGG16	-0.636	0.516	0.685
resnet18	-0.715	0.192	0.696
resnet50	-0.7172	0.053	0.6677
resnet152	-0.7158	-0.035	0.7228
densenet121	-0.726	-0.015	0.6732

TABLE 3
The result of the **Imagenet** validation dataset. This table shows the Pearson correlation coefficient between different geometric features and model magnitude pruning (*supplementary equation (2)*). Angle and margin show a significant correlation with pruning vulnerability. The correlation between l2 and pruning vulnerability is subtle.

CNN Architecture	angle	l2	margin.
VGG16	-0.6739	0.3924	0.6534
VGG19	-0.6667	0.3984	0.6514
resnet18	-0.6795	0.1737	0.7282
resnet50	-0.6933	0.083	0.7058
Densenet121	-0.6816	0.0252	0.7154

used as a metric to highlight vulnerable and resilient samples without comparing many models, and it is a significant advantage compared with the previous approach [7].

Model pruning and retraining not only recover the performance of the pruned model but also improve it, especially for certain data samples. We can check the feature attribution heat map to the margin in Fig. 10. We compare the geometric feature attribution heat map of the pre-trained model and retrained model with 50% weight removed. We select Fig. 10(①) samples (green color) that are predicted incorrectly in the pre-trained model but correctly in the retrained model. The geometric feature attribution visualization in Fig. 10(③) shows that the retrained model captures better features than the pre-trained model.

7.4 User Evaluation

We performed a qualitative user study with machine learning users to further evaluate the visualization system. The interview involves four machine learning researchers (E1, E2, E3, E4) with a solid neural network model background. E1-E4 are graduate-level machine learning researchers who are not engaged in the visualization system design of this article and are not co-authors of this paper. Two of them are

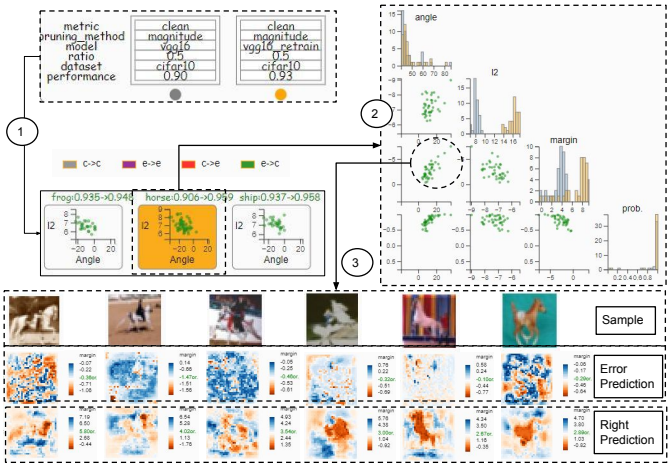

Fig. 10. Model pruning and retraining can improve prediction performance. Comparing the retrained and original model (①), the samples that are predicted wrong in the original model but predicted right in the retrained model have captured better features to margin. (③).

senior researchers (E1, E2) ($\geq$ 6 years of machine learning experience), and the other two are juniors (E3, E4) ($\geq$ 3 years of machine learning experience). They share the common requirement for model and sample comparison in their daily research which is also the key requirement of the design system. All of them have experience in performing model compression (e.g., network pruning), and how to perform input perturbation to evaluate model prediction robustness. The evaluation also includes comments from E5 who is involved in the initial system design process and is a co-author of this paper. During the discussion, E5's comments are separated from the comments of E1-E4.

We introduce basic concepts during the interview before presenting the system to the users. Then, we introduce the visual interface, explain how different views interact, and present three use cases in this paper. We leave time for users to play around with the system (all these steps take 30-40 minutes). After the demonstration, we ask users to perform the critical basic tasks of this system - How to perform pruning/benchmark comparison, how to perform the model comparison, and how to reveal the impact of pruning and data corruption on samples. Such an operation would help us understand the system users' learning curve. At the end of the study, we discuss with users about the visualization system from positive and negative perspectives and future work. The step takes around 25 to 30 minutes and the whole process takes around 70 minutes.

7.4.1 Usefulness

The learning curve for users to understand the visualization system depends on the users' background. During the interview, interviewees could finish the designed task we expected them to perform after the system presentation. However, the system may be challenging for new users unfamiliar with dimension reduction techniques like UMAP, and the concept of feature attribution. Here, we summarize key comments and feedback from users on the system's properties to accelerate domain exploration and motivate their research.

A scalable model comparison and multi-level micro-modification analysis system. The multi-level comparison analysis from pruning/evaluation, model, and sample gives users a progressive way to query the information visually. E1 stated, "...I like the hierarchical process of this system in understanding the model. I can imagine that this system can be very helpful in understanding the high-risk application...". E2 stated, "...One of the most positive sides of this visualization system is the model embedding. It gives a global view of all model's similarities and also details comparison between models...".

Users also mention that the scalable model comparison is useful for pruning analysis and beyond. As E3 stated, "...The usability of the visualization is beyond the scope of pruning but can be used in many different domains...". As stated by domain experts, similar to pruning, hyper-parameter tuning, sample subset selection, and different architecture comparisons also involve a large amount of model comparison and also benefit from scalable model comparison.

Data quality tracker. The users mention that the current layout provides a convenient interface for quickly and successfully locating the challenging subset of samples for the pruned models, revealing the evaluation's efficiency. As E4 stated, "..the system provides an ability of data quality backtracking, which helps me identify which model improves these outlier samples' performance and which model does not. It reveals the problem of datasets if they are unrepresentative and if all models perform badly. During the data collection process in their application, it is helpful to locate a set of underrepresented samples and add more of these types of samples to the training dataset to improve the model's performance. They can use the system to iteratively decide the next subset of samples they need to collect...". The system also provides interactive visualization to support the data evaluation benchmark comparison and tells their similarity, as demonstrated in Section 7.1. E1 stated, "...How do we select the representative benchmark to evaluate the robustness? Do we need more benchmarks, and which benchmark is most useful? The current system provides a visual tool for me to identify them...".

We also include comments from an expert collaborator (E5). "The most impressive attribute of this system is that it can compare different pruning methods on different original models at the same time. Its standout feature lies in its ability to simultaneously compare various pruning methods across original models. Through uniform geometric methods, it enables comprehensive comparisons of pruning methods across diverse model architectures, sparsity levels, and data noise".

7.4.2 Reflection and Future Works

During our interview, users also point out some limitations of the current visualization system.

Lack of generalization and miss functionalities The geometric comparison methodology is useful in classification tasks but fails to be generalized to other tasks such as object detection and image segmentation. During our interview, users also mentioned that the current visualization system lacks the ability to identify the bias caused by pruning operations. Pruning can be a potential way to introduce additional bias into the model, and the current framework has few components to support this analysis.

The scalability of visualization. Users mention that the system provides rich information to analyze the model pruning and prediction robustness. However, such a design also brings dense information into the analysis process. One additional potential limitation is the scalability of the global geometric view and local geometric view. As for more complex datasets, such as Imagenet (1000 classes), visualizing all *class directions* at the same is not practical. Accordingly, a preselection or ranking operation can greatly mitigate this challenge.

Furthermore, beyond the geometric proper, we can think about using general metrics such as CKA [58], a basic metric to scalablely construct the model comparison. Another potential improvement of the current system for future work comes from additional introspection ability, i.e., can we combine other model explanation tools such as concept-based explanation [59], [60] with the proposed geometry metric, to better articulate the exact semantics changes induced by pruning?

8 Conclusion

In this work, we coordinate geometrically inspired metrics in the neural network latent space for a comparative study of how widely adopted (and state-of-the-art) model pruning approaches impact neural network models' internal representation and prediction robustness. Our study reveals that the geometric location of a sample in the high-dimensional feature representation space is critical for a model's prediction. The proposed visualization system compares the similarity among different robustness evaluation benchmarks, provides valuable insights for explaining model robustness from a geometric perspective, and visually reveals the difference among pruning methods that produce surprisingly robust models whereas others reduce model robustness.

References

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[2] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko *et al.*, "Highly accurate protein structure prediction with alphafold," *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.

[3] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. Nelson, A. Bridgland *et al.*, "Improved protein structure prediction using potentials from deep learning," *Nature*, vol. 577, no. 7792, pp. 706–710, 2020.

[4] Y. Huang, Y. Cheng, A. Bapna, O. Firat, D. Chen, M. Chen, H. Lee, J. Ngiam, Q. V. Le, Y. Wu *et al.*, "Gpipe: Efficient training of giant neural networks using pipeline parallelism," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[5] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[6] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2736–2744.

[7] S. Hooker, A. Courville, G. Clark, Y. Dauphin, and A. Frome, "What do compressed deep neural networks forget?" *arXiv preprint arXiv:1911.05248*, 2019.

[8] L. Liebenwein, C. Baykal, B. Carter, D. Gifford, and D. Rus, "Lost in pruning: The effects of pruning neural networks beyond test accuracy," *Proceedings of Machine Learning and Systems*, vol. 3, pp. 93–138, 2021.

[9] S. Bulusu, B. Kailkhura, B. Li, P. K. Varshney, and D. Song, "Anomalous example detection in deep learning: A survey," *IEEE Access*, vol. 8, pp. 132 330–132 347, 2020.

[10] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," in *International Conference on Learning Representations*, 2018.

[11] J. Diffenderfer, B. R. Bartoldson, S. Chaganti, J. Zhang, and B. Kailkhura, "A winning hand: compressing deep networks can improve out-of-distribution robustness," in *Proceedings of the 35th International Conference on Neural Information Processing Systems*, ser. NIPS '21, 2024.

[12] J. Diffenderfer and B. Kailkhura, "Multi-prize lottery ticket hypothesis: Finding accurate binary neural networks by pruning a randomly weighted network," in *International Conference on Learning Representations*.

[13] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[14] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[15] Y. LeCun, J. Denker, and S. Solla, "Optimal brain damage," *Advances in Neural Information Processing Systems*, vol. 2, 1989.

[16] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *7th International Conference on Learning Representations, ICLR, New Orleans, LA, USA, May 6-9*, 2019.

[17] V. Ramanujan, M. Wortsman, A. Kembhavi, A. Farhadi, and M. Rastegari, "What's hidden in a randomly weighted neural network?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 893–11 902.

[18] G. Li, J. Wang, H.-W. Shen, K. Chen, G. Shan, and Z. Lu, "Cnnpruner: Pruning convolutional neural networks with visual analytics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1364–1373, 2020.

[19] J. Wang, L. Gou, W. Zhang, H. Yang, and H.-W. Shen, "Deepvid: Deep visual interpretation and diagnosis for image classifiers via knowledge distillation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 6, pp. 2168–2180, 2019.

[20] F. Hohman, H. Park, C. Robinson, and D. H. P. Chau, "S ummit: Scaling deep learning interpretability by visualizing activation and attribution summarizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 1096–1106, 2019.

[21] M. Liu, S. Liu, H. Su, K. Cao, and J. Zhu, "Analyzing the noise robustness of deep neural networks," in *2018 IEEE Conference on Visual Analytics Science and Technology (VAST)*, 2018, pp. 60–71.

[22] M. Kahng, P. Y. Andrews, A. Kalro, and D. H. Chau, "A cti v is: Visual exploration of industry-scale deep neural network models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 88–97, 2017.

[23] J. Sun, A. Mehra, B. Kailkhura, P.-Y. Chen, D. Hendrycks, J. Hamm, and Z. M. Mao, "Certified adversarial defenses meet out-of-distribution corruptions: Benchmarking robustness and simple baselines," *arXiv preprint arXiv:2112.00659*, 2021.

[24] E. Mintun, A. Kirillov, and S. Xie, "On interaction between augmentations and corruptions in natural corruption robustness," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[25] C. Michaelis, B. Mitzkus, R. Geirhos, E. Rusak, O. Bringmann, A. S. Ecker, M. Bethge, and W. Brendel, "Benchmarking robustness in object detection: Autonomous driving when winter is coming," *arXiv preprint arXiv:1907.07484*, 2019.

[26] J. Sun, Q. Zhang, B. Kailkhura, Z. Yu, C. Xiao, Z. M. Mao, A. Goyal, H. Law, B. Liu, A. Newell *et al.*, "Benchmarking robustness of 3d point cloud recognition against common corruptions," in *International Conference on Machine Learning*, 2021.

[27] D. Ren, S. Amershi, B. Lee, J. Suh, and J. D. Williams, "Squares: Supporting interactive performance analysis for multiclass classifiers," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 61–70, 2016.

[28] J. Zhang, Y. Wang, P. Molino, L. Li, and D. S. Ebert, "Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 364–373, 2018.

[29] A. Hinterreiter, P. Ruch, H. Stitz, M. Ennemoser, J. Bernard, H. Strobelt, and M. Streit, "Confusionflow: A model-agnostic visualization for temporal analysis of classifier confusion," *IEEE Transactions on Visualization and Computer Graphics*, 2020.

[30] A. Chatzimparmpas, R. M. Martins, K. Kucher, and A. Kerren, "Stackgenvis: Alignment of data, algorithms, and models for stacking ensemble learning using performance metrics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1547–1557, 2020.

[31] J. Wang, L. Wang, Y. Zheng, C.-C. M. Yeh, S. Jain, and W. Zhang, "Learning-from-disagreement: A model comparison and visual analytics framework," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[32] Y. Li, T. Fujiwara, Y. K. Choi, K. K. Kim, and K.-L. Ma, "A visual analytics system for multi-model comparison on clinical data predictions," *Visual Informatics*, vol. 4, no. 2, pp. 122–131, 2020.

[33] D. Cashman, A. Perer, R. Chang, and H. Strobelt, "Ablate, variate, and contemplate: Visual analytics for discovering neural architectures," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 863–873, 2019.

[34] J. Yuan, C. Chen, W. Yang, M. Liu, J. Xia, and S. Liu, "A survey of visual analytics techniques for machine learning," *Computational Visual Media*, vol. 7, pp. 3–36, 2021.

[35] J. Wang, S. Liu, and W. Zhang, "Visual analytics for machine learning: A data perspective survey," *IEEE Transactions on Visualization and Computer Graphics*, 2024.

[36] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1321–1330.

[37] J. Xia, Y. Zhang, J. Song, Y. Chen, Y. Wang, and S. Liu, "Revisiting dimensionality reduction techniques for visual cluster analysis: An empirical study," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 1, pp. 529–539, 2021.

[38] V. Sivaraman, Y. Wu, and A. Perer, "Emblaze: Illuminating machine learning representations through interactive comparison of embedding spaces," in *Proceedings of the 27th International Conference on Intelligent User Interfaces*, 2022, pp. 418–432.

[39] A. Boggust, B. Carter, and A. Satyanarayan, "Embedding comparator: Visualizing differences in global structure and local neighborhoods via small multiples," in *Proceedings of the 27th International Conference on Intelligent User Interfaces*, 2022, pp. 746–766.

[40] Z. J. Wang, F. Hohman, and D. H. Chau, "Wizmap: Scalable interactive visualization for exploring large machine learning embeddings," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, 2023, pp. 516–523.

[41] S. Liu, B. Wang, P.-T. Bremer, and V. Pascucci, "Distortion-guided structure-driven interactive exploration of high-dimensional data," in *Computer Graphics Forum*, vol. 33, no. 3. Wiley Online Library, 2014, pp. 101–110.

[42] D. Blalock, J. J. Gonzalez Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" *Proceedings of Machine Learning and Systems*, vol. 2, pp. 129–146, 2020.

[43] N. Lee, T. Ajanthan, and P. H. Torr, "Snip: Single-shot network pruning based on connection sensitivity," *ICLR*, 2019.

[44] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *CVPR*, 2019, pp. 11 264–11 272.

[45] B. Hassibi and D. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," *Advances in Neural Information Processing Systems*, vol. 5, 1992.

[46] K. Sreenivasan, S. Rajput, J.-y. Sohn, and D. Papailiopoulos, "Finding nearly everything within random binary networks," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022, pp. 3531–3541.
[47] W. Liu, Z. Liu, Z. Yu, B. Dai, R. Lin, Y. Wang, J. M. Rehg, and L. Song, "Decoupled networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2771–2779.
[48] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, p. 507–516.
[49] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *The craft of information visualization*. Elsevier, 2003, pp. 364–371.
[50] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," in *International Conference on Learning Representations*.
[51] M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh, "Beyond accuracy: Behavioral testing of nlp models with checklist," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 4902–4912.
[52] J. Yuan, M. Liu, F. Tian, and S. Liu, "Visual analysis of neural architecture spaces for summarizing design principles," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 1, pp. 288–298, 2023.
[53] L. McInnes, J. Healy, N. Saul, and L. Großberger, "Umap: Uniform manifold approximation and projection," *Journal of Open Source Software*, vol. 3, no. 29, 2018.
[54] P. Xenopoulos, J. Rulff, L. G. Nonato, B. Barr, and C. Silva, "Calibrate: Interactive analysis of probabilistic model output," *IEEE Transactions on Visualization and Computer Graphics*, 2022.
[55] A. Adadi and M. Berrada, "Peeking inside the black-box: a survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.
[56] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.
[57] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, "Sanity checks for saliency maps," *Advances in Neural Information Processing Systems*, vol. 31, 2018.
[58] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," in *International Conference on Machine Learning*. PMLR, 2019, pp. 3519–3529.
[59] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas *et al.*, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2668–2677.
[60] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 618–626.