Abstract

Understanding the black-box representations in Deep Neural Networks (DNN) is an essential problem in deep learning. In this work, we propose Graph-Based Similarity (GBS) to measure the similarity of layer features. Contrary to previous works that compute the similarity directly on the feature maps, GBS measures the correlation based on the graph constructed with hidden layer outputs. By treating each input sample as a node and the corresponding layer output similarity as edges, we construct the graph of DNN representations for each layer. The similarity between graphs of layers identifies the correspondences between representations of models trained in different datasets and initializations. We demonstrate and prove the invariance property of GBS, including invariance to orthogonal transformation and invariance to isotropic scaling, and compare GBS with CKA. GBS shows state-of-the-art performance in reflecting the similarity and provides insights on explaining the adversarial sample behavior on the hidden layer space.

1. Introduction

It is indisputable that DNN has achieved the most advanced performance in the fields like computer vision [33], natural language processing [4], etc. In most machine learning tasks, a Deep Neural Network (DNN) learns feature representations end to end. Despite great success, how to understand and characterize the presentations DNN learns from data is still under-explored [21]. The understanding and interpretation of DNN representations can help to determine whether a learned DNN is applicable to various domains and the transferability to a new problem [57]. We can also compare models with the same structure trained on different data, or different structures and training strategies, to unveil what causes the model to behave differently.

Recent studies reveal the difference in the representations of models with different structures by comparing their similarities. Raghu et al. [31] analyzed the internal representation structure of the most advanced vision models, including vision transformer [9], ResNet [18], and MLP-Mixer [41]. Nguyen et al. [28] analyzed the effect of architecture depth and width on learned representations. The similarity index they used is Centred Kernel Alignment [6, 21] (CKA), which has been thought [21] the best similarity measurement among all the other similarity indexes. Kornblith et al. [21] discussed the invariance properties of similarity indexes, including invariance to Invertible Linear Transformation (ILT), Invariance to Orthogonal Transformation (OT), and invariance to Isotropic Scaling (IS). They also proved that CKA is better than other indexes through a sanity check, i.e., using the similarity index to identify the corresponding layer of models of different initialization trained with the same structure and dataset. The layers at the same depth should have the highest similarity index.

In this work, we aim to develop a method to measure the similarity between different DNN representations. Inspired by the works of graphs in network topology modeling [44] and the graph-based knowledge embedding [51], we propose Graph-Based Similarity (GBS), which measures the representation similarity based on the graph of the layer features. Specifically, a batch of samples with a balanced distribution of labels are fed into the target model to obtain the features of each layer, then we calculate the cosine similarity of each layer with respect to its feature map of different inputs. The cosine similarity result is used to construct the adjacent matrix, where each input is the node and the corresponding similarity is the edge weight. In this way, the learned representations of different categories are fully expressed in the graph, while hidden layer features of different shapes can also be compared since the graph shape is only related to the input batch size.

Through experiments, we observed that GBS provides a meaningful correspondence index between layers of the same model and different models, across different datasets. We further show that GBS is empirically a better similarity

\footnote{Code at: https://github.com/implicitDeclaration/similarity}
index compared with CKA. The pathology of a learned network is reflected in the similarity matrix of different layers. We also observed that adversarial samples greatly affect the hidden layer representations, especially in the deep layers.

Our contribution are summarized as follows:

1. We propose Graph-Based Similarity (GBS) as a DNN representation similarity index to identify the correspondence across DNN representation. GBS is invariant to orthogonal transformation and isotropic scaling but not invertible linear transformation.

2. We show that GBS is a good index to determine the correspondence between the hidden layers of different depth, width, and with different tasks. It outperforms CKA in the sanity check of finding the corresponding layer of the same structural model.

3. We demonstrate that GBS can be used to reveal the pathology of networks and provides insights for understanding the adversarial sample’s behavior in the hidden layer space.

The rest of paper is organized as follows. Sec. 2 presents the related work, including other similarity indexes, how to identify a good similarity index, and related works on interpreting DNN representations. Sec. 4 introduces how GBS works and provides proof of the invariance property of GBS. Sec. 5 gives the experimental setup and results. Sec. 6 concludes the paper and Sec. 7 describes future works.

2. Related Works

Recent works proposed many similarity indexes and what attributes a good index should have. There are also works dedicated to interpreting DNN representations and adversarial attacks in a way other than similarity index.

2.1. Similarity Index

Raghu et al. [30] proposed Singular Vector Canonical Correlation Analysis (SVCCA) for comparing two representations. They defined a neuron as a vector in $R^m$ over a dataset with $m$ examples. A layer consisting of multiple neurons can be considered as the subspace $l$ of $R^m$ spanned by its neurons. SVCCA takes two sets of neurons $l_1$ and $l_2$ as input, and performs a singular value decomposition of each subspace to get sub-subspaces $l'_1$ and $l'_2$. Then SVCCA computes the Canonical Correlation similarity [15] of the obtained sub-subspaces. The output of CCA is a set of coefficients corresponding to vectors of the SVD result. Morcos et al. [27] further improved the CCA method. They argued that SVCCA suffers from not effectively distinguishing between the signal and the noise in the representation since SVCCA implicitly assumes that all CCA coefficients are equally important. Morcos et al. showed that the coefficients and their corresponding vectors may be noise. They addressed the problem by replacing the mean with a weighted mean.

Kornblith et al. [21] analyzed different similarity indexes and concluded that CKA [6] is a better similarity index compared with CCA based indexes. They argued that the similarity index should be invariant to orthogonal transformation and isotropic scaling, but not invertible linear transformation. Because neither CCA nor other statistics that are invariant to ILT can give a meaningful similarity measurement between representations that have higher dimensions than data points.

Tang et al. [40] proposed that a DNN model representation should contain both feature vectors and gradients. They use two kernel matrices for the feature and gradient, respectively, and combine them using the Hadamard product.

2.2. How to identify a good similarity indicator?

As argued by Kornblith et al. [21], similarity should be invariant to OT and IS but not ILT. A similarity index $s$ is invariant to ILT if $s(X, Y) = s(XA, YB)$ for any full rank $A$ and $B$, where $X$ and $Y$ denote the activation outputs. ILT indicates that the scale of directions in activation space is irrelevant to the index. As mentioned and proved in [21], if a similarity index is invariant to ILT, it will give the same result for any representation of width greater than or equal to the dataset size. This situation widely exists in convolutional networks [24,38]. The training of a DNN is either invariant to ILT, which is demonstrated by the popularity of batch normalization [20]. Invariance to OT is a weaker requirement compared with ILT. Invariant to OT means $s(X, Y) = s(XU, YV)$ for any full-rank orthonormal matrices $U$ and $V$. It implies that the index is invariant to permutation, which is necessary considering the symmetries of neural networks [29]. Invariant to IS asks for $s(X, Y) = s(\alpha X, \beta Y)$ for any $\alpha, \beta \in \mathbb{R}^+$, i.e., the index is not affected by rescaling of individual features. We refer interested readers to [21] for more details. Tang et al. [40] gave three empirical indicators when comparing different similarity indexes. The factors they considered include model structure, initialization, datasets, the choice of layer, and the number of categories in the task. They argued that under the same other conditions, a good method should assign a higher similarity score for 1) models trained on the same dataset than trained on different datasets; 2) models trained on the similar datasets (CIFAR10 and CIFAR100) than trained on not similar datasets (CIFAR10 and SVHN); 3) models trained with the same number of categories than trained with the different number of categories.

We show the invariant property of GBS in Sec. 4 and prove that GBS is empirically a better similarity measurement in Sec. 4.
2.3. Interpreting DNN Representations

In addition to seeking for a similarity index, previous works also explored other ways to interpret DNN representations \([7][26][27][43]\), e.g., visualizing the hidden layer representations \([34][35]\), reasoning explanation \([11][14]\), and gradient-based attribution methods \([2][35]\). Visualization-based interpretation \([45]\) aims to explain the model decision by depicting the correlation between input space to the final output. In other words, it finds the area in the input images that determines the DNN decisions. The area is supposed to contain the target object while excluding the background. Ge et al. \([11]\) provided new insight on interpreting the model decision by their visual reasoning explanation framework (VRX). VRX uses high-level category-specific visual concepts to decompose the input object. It gives an understanding of the model inference process by demonstrating the structural relationships between these visual concepts.

Compared with other interpreting methods, a similarity index focusing on the model level comparison, instead of a single input image. Though a similarity index is not as intuitive as a visualized result, we can still obtain valuable information by further analysis \([28][31]\).

2.4. Explaining Adversarial Attacks

In this work we use GBS to reveal how adversarial samples affect the DNN representation. For an image classification model, adversarial sample is a kind of images that added with human imperceptible perturbations, that can deceive the classifier \([39]\). Previous works proposed explanations for adversarial samples, such as the linear nature of the DNN model \([13]\) and the features of datasets \([19]\), but there is a lack of consensus on the explanation \([1]\). Many recent works explain adversarial samples in different perspectives. Zhang et al. \([47]\) proposed to understand adversarial samples by disentangling the clean images and adversarial perturbations. They argued that for an example generated by universal adversarial attacks, the perturbation has the dominant feature. Zhang et al. \([48]\) explain the adversarial robustness with neuron sensitivity. They argue that sensitive neurons make the most non-trivial contributions to the misclassification of adversarial samples. Guo et al. \([15]\) found that the transferability of adversarial samples can be enhanced by improving the linearity of DNNs. In this work, we analyze the adversarial sample behavior in the internal representation of DNN using GBS.

3. Method

As shown in Fig. 1, GBS requires a batch of samples to obtain the hidden layer representations. After that, representations of a layer are embedded in a graph, then we compare the similarity between graphs as the similarity of layer representations.

3.1. Graph-Based Similarity

Given a set of inputs \(\mathcal{X} = \{x_1, x_2, \ldots, x_N\}\) where \(N\) represents the number of inputs derived from the dataset, we feed them into the target model with fixed parameters to obtain neural network hidden representations \(F_i \in \mathbb{R}^{N \times C_i \times W_i \times H_i}\), \(i = 1, \ldots, L\), where \(L\) is the number of layers, \(C_i\) is the number of channels in \(i\)-th layer, and \(W_i\) and \(H_i\) are width and height of feature maps in \(i\)-th layer, respectively. Note that we treat a building block containing multiple convolutional layers as a layer for ResNet \([18]\) and treat a single convolutional layer as a layer for VG-GNet \([36]\). Then we reshape the obtained representation to reduce its dimension, i.e., \(\mathbb{R}^{N \times C_i \times W_i \times H_i} \rightarrow \mathbb{R}^{N \times M_i}\), where \(M_i = C_i \times W_i \times H_i\).

For \(i\)th layer of the target model, the network hidden representations \(F_i^N\) of inputs \(\mathcal{X}\) is \(F_i^N = \{F_i^1, F_i^2, \ldots, F_i^N\}\). Since a useful DNN learns from data, starting from random weights and modifying its weights according to the data and
the similarity is calculated by

\[ e_{nm} = \frac{F^n_i \cdot F^m_i}{\|F^n_i\|_2 \cdot \|F^m_i\|_2}, \]

which is widely used to quantify the similarity between two non-zero vectors \( v \).

The number of nodes in \( G_i \) is determined by the number of inputs \( N \), which is independent of the feature map size of the layer. Then we obtain the similarity of layer \( i \) and layer \( j \) by computing the similarity \( S_{ij} \) between \( G_i \) and \( G_j \). Layer \( i \) and layer \( j \) can be any of the different models or layers at different depths of the same model. We use Layer Similarity [49] (LSim) to calculate the similarity between layers. LSim are proposed to measure the similarity between layers of complex and multiplex network. Assume the adjacent matrix of \( G_i \) and \( G_j \) are \( A_i \) and \( A_j \), respectively, the similarity is calculated by

\[ S_{ij} = \frac{1}{N} \sum_{k=1}^{N} f_{\cos}(A_{i,k}, A_{j,k}), \]

where \( A_{i,k} \) is the row vector of \( A_i \) and \( f_{\cos}(\cdot) \) is the cosine similarity between two vectors, which is

\[ f_{\cos}(D_1, D_2) = \frac{D_1 \cdot D_2}{\|D_1\|_2 \cdot \|D_2\|_2}. \]

However, when \( N \) is large, a fully connected graph will induce noise to the similarity result, since many irrelevant nodes are also connected. To obtain a better result, we only connect the top \( k \) nodes (\( k \) equals to the degree of the graph) with the highest similarity at the step of graph construction.

### 3.2. Invariance Property of GBS

GBS first constructs the representation graph, then calculates the similarity between graphs. The graph \( G = (V, E) \) is defined by its nodes and edges. In Eq. (2), we use an adjacent matrix to calculate the similarity, which is related to the number of nodes \( N \) and the edge weights. \( N \) only depends on the number of inputs that is irrelevant to the transformations on representations. Thus edge weights are the only factor that affects GBS. In the step of constructing graphs, edge weights are computed by Eq. (1).

GBS is invariant to OT and IS but not ILT. The edge weight after ILT is

\[ ILT(e_{nm}) = \frac{A F^n_i \cdot F^m_i}{\|A F^n_i\|_2 \cdot \|F^m_i\|_2} \neq e_{nm}, \]

where \( A \) is an arbitrary full rank matrix with the same size of \( F \).

According to the definition of OT, it preserves a symmetric inner product, i.e., preserving lengths of vectors and angles between vectors. For two vectors \( u \) and \( v \), we have

\[ \|OT(v)\| = \|v\| < u, v > \leq < OT(u), OT(v) >. \]

The edges are weighted with cosine similarity using Eq. (1), which is only relevant to the angles between the two vectors. Since OT does not change the angles between vectors, GBS is invariant to OT.

IS is the rescaling of representations, i.e., the same scales are applied to each dimension. After IS, a vector \( v \) can be expressed as \( \alpha v \), where \( \alpha \in \mathbb{R}^+ \). The weight of the edge between two nodes becomes

\[ IS(e_{nm}) = \frac{\alpha F^n_i \cdot F^m_i}{\|\alpha F^n_i\|_2 \cdot \|F^m_i\|_2} = e_{nm}. \]

Since the graph edges are not affected by IS, GBS is invariant to IS.

### 3.3. Extensibility of GBS

In our work, we adopt LSim to measure the similarity of graphs that correspond to interval representations. There are other graph similarity measurements in addition to LSim, e.g., degree correlation [3].

The degree of a node is defined as the number of edges connected to it. Assume there is \( \alpha = 1, 2, ..., N \) nodes in layer \( i \) and the degree of node \( \alpha \) is \( k^{i}_{\alpha} \). The degrees of the same node in different layers can be correlated and reflects the corresponding representation similarity between the two layers. Assume the degree sequence of layer \( i \) is \( k^i \), which is a \( 1 \times N \) vector, then we can use correlations like Pearson. For example, the Pearson correlation of layer \( i \) and \( j \) using degree sequence is

\[ \rho_{i,j} = \frac{E[(k^i - \mu_i)(k^j - \mu_j)]}{\sigma_i \sigma_j}, \]

where \( \mu_i \) is the mean value of \( k^i \) and \( \sigma_i \) is the variance of \( k^i \). Note that Kendall and Spearman correlation are not applicable here since they converts the original degrees into their corresponding ranks. The ranks are all the same in \( k^i \).

We use LSim since it shows a good result in the preliminary experiments. We show the result of using other correlation indexes in the appendix.
4. Experiments

We show that GBS is an informative similarity index in this section and compare GBS with CKA. By the ablation analysis and a sanity check, we demonstrate that GBS is robust to the choice of graph size and degree. Then we show the similarity of all the layers in the same model trained with different datasets. The similarity of the representation between layers reflects its structural information. We further show the layer similarity of different initialization and datasets. Finally, we show the behavior in the feature space when adversarial sample transfer to different models using GBS.

4.1. Experimental Settings

We conduct our experiments on CIFAR10, CIFAR100 [22], and ImageNet [8]. The three datasets contain 10, 100, and 1000 categories of images, respectively. The images size of CIFAR10 and CIFAR100 is $32 \times 32$ while ImageNet is $224 \times 224$. The networks we use are VGGNet [36] and ResNet [18].

4.2. Ablation Analysis and Sanity Check

Our ablation analysis focus on the influence of 1) the number of inputs (graph size) and 2) the number of connected nodes (degree of the graph). We use a sanity check to evaluate the performance of GBS, which is also adopted in the work of Kornblith et al. [21]. Specifically, given a pair of architecturally identical models trained with the same dataset but different random initialization, the similarity index should assign the highest score for the architecturally corresponding layers. We use ten ResNet18 and VGGNet16 of different initialization trained on CIFAR10, and compute the average accuracy over all the layers. The initialization utilizes Kaiming normalization [17] with ten different random seeds. The accuracy of all trained models is above 92%.

Given the intuition that an adequate number of nodes is beneficial to obtain a good similarity, we start with analyzing the influence of degree under a fixed number of nodes (500). The 500 samples are randomly selected from the dataset that has a balanced label distribution. The result is shown in Fig. 2. The degree value varies from 1 to 10. It shows that the degree of the generated graph has little impact on finding the layer correspondence. Even when the degree is set to 1, the accuracy only drops less than 6% compared with the degree set to 10. Considering the computational cost, we set the degree to 5, i.e., only connect the top 5 most correlated nodes in the graph.

Then we evaluate the performance of GBS with the graph size (number of nodes in the graph) varies from 5 to 500. Note that the number of categories of CIFAR10 is 10. When the graph size is set to 5, i.e., less than the number of categories of CIFAR10, GBS still achieves accuracy over 61.0% and 84.4% on VGGNet16 and ResNet18, respectively. When the graph size is greater than 50, the accuracy does not change with it basically.

In summary, the ablation analysis indicates that GBS is robust to the choice of graph size and degree. A large number of nodes (input batch size) and large degree values slightly help increase the performance of GBS. However, considering the computational cost, excessively large number of nodes and large degree values are unnecessary.

We also compare CKA and GBS in Fig. 2. CKA uses a linear kernel and the input batch is set to 500, as suggested in [21] that RBF and linear kernels give similar results across most experiments. The dash lines denote the accuracy of CKA. We find that GBS outperforms CKA in identifying the correspondence of complex CNNs, like VGGNet and ResNet.

4.3. Similarity of Model Layers

GBS reveals the pathology in neural network representations. As shown in Fig. 3 we use the confusion matrix to demonstrate the similarity of different layers within a model, where the x-axis and y-axis represent layers of the model. The models in Fig. 3 are trained on CIFAR10, we also provide the results of CIFAR100 and ImageNet in the supplementary material. The detailed model structure of VGGNets and ResNets used in the experiment are shown in the supplementary material.
For both VGGNets and ResNets, the grid in the confusion matrix exactly corresponds to the size and depth of the block in the model, i.e., the similarity of layers inside the same block is higher than layers in different blocks. Within a grid, the color gradually darkens with the diagonal as the center, which means inside the same block, the greater the distance between layers, the lower the similarity. Because within a block, DNN extracts more abstract features layer by layer [46], the similarity gradually decreases accordingly. The residual connections deliver the features of different depth between the entry of blocks, making the representations of the current block significantly different from the post block.

Even if trained on the same dataset, ResNet of different structures learns different representations. Block size has a significant effect on the learned representations. We compare the similarity between the first layers of each block of ResNets, as shown in Fig. 4. In shallow blocks, models with approximate size have more similar representations. However, as the layer goes deeper, the difference of learned representations grows, making the deep features different.

### 4.4. Similarity of Independently Trained Models

With the same model structure and dataset, initialization of different distributions mainly affects the learned representations in deep layers. A good weight initialization makes the output of the convolution and the gradient of the forward pass stable when propagating forward or backward. We evaluate two kinds of most widely used initialization, i.e., Kaiming initialization [17] and Xavier initialization [18]. Xavier initialization assumes that the activation function is linear, e.g. Tanh activation function. It argued that, to avoid the degeneration of gradients, the output value of each layer should maintain a Gaussian distribution. It set

---

**Table 1. VGGNets Configuration.**

| VGG11 | VGG13 | VGG16 | VGG19 |
|-------|-------|-------|-------|
| conv3-64 | conv3-64 x 2 | conv3-64 x 2 | conv3-64 x 2 |
| conv3-128 | conv3-128 x 2 | conv3-128 x 2 | conv3-128 x 2 |
| conv3-256 x 2 | conv3-256 x 2 | conv3-256 x 3 | conv3-256 x 2 |
| conv3-512 x 2 | conv3-512 x 2 | conv3-512 x 3 | conv3-512 x 2 |

Figure 3. The similarity between the layers of VGGNets and ResNets on CIFAR10.

Figure 4. The similarity between the blocks of different ResNets on CIFAR10. There are totally 4 blocks in ResNet, Block1 to Block4 are ordered from shallow to deep.
the biases to be 0 and the weights $W_{ij}$ with

$$W_{ij} \sim U[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}],$$

(8)

where $U[-a, a]$ is the uniform distribution in the interval $(-a, a)$ and $n$ is the size of the previous layer. Kaiming initialization argued that since most DNNs are constructed with non-linear activation functions, e.g., ReLu activation function, the assumption in Xavier initialization is invalid. In this case, the initialization of weights should be a zero-mean Gaussian distribution whose standard deviation is $\frac{\sqrt{2}}{n}$. Specifically, there are four initialization methods in the experiment, including Kaiming Normal, Kaiming Uniform, Xavier Normal, and Xavier Constant, as shown in Fig. 5. Though Block 1 shows a high similarity between all kinds of distributions, the remaining blocks are different from each other, and the similarity can be as low as 0.64. It indicates that deep layers learn different representations under the initialization of different distributions.

### 4.5. Adversarial Samples in Hidden Layer Representation

To analyze whether the internal representations are similar when the attack transfers to different models, we generate adversarial attacks with one model and attack other models with the same batch of samples. The models used here are VGGNets of different initialization. We use `advertorch` package to generate untargeted adversarial samples for the VGGNet16 model trained on CIFAR10. The adversarial attacks include Fast Gradient Sign Method [13] (FGSM), Carlini & Wagner Attack [5] (CW), Basic Iterative Method [23] (BIM), and Project Gradient Descent [25] (PGD). All the attacks use the default parameters and the attack success rates of PGD, BIM, FGSM, and CW for the original model are 56%, 50%, 41%, and 54%, respectively. The batch size is set to 100. As shown in Fig. 6a, compared with the normal samples, all attacks show that the similarity decreases with the increase of depth. When the same batch of normal samples are fed into other models, as shown in Fig. 6b, the representations show relatively high similarity at all depths. However, when fed with the same batch of adversarial samples, the adversarial samples successfully fool them with different representations activated in the other three models. It indicates that the transferability of adversarial samples is not because the same features are activated.

We show the similarity between different types of adversarial examples in the feature space, as shown in Fig. 7. We calculate the similarity between different adversarial attacks every 3 layers in VGGNet16. Targeted attacks and untargeted attacks are compared, and the similarity is calculated using cosine similarity.
Figure 6. Representation similarity of adversarial samples transferring to different models. In (a), the similarity is calculated with the same original model that input normal samples. In (b), (c), and (d), the similarity are calculated with the layer of other models and the corresponding layer in the original model.

Figure 7. The similarity of different adversarial attacks.

5. Conclusion

In this paper, we propose Graph-Based Similarity (GBS) to measure the similarity between representations of DNN. GBS can identify the correspondence of representations across hidden layers of different sizes and depths. We prove that GBS is a better similarity index than CKA through a sanity check. It is robust to changes in graph size and degree. By using GBS, we find that the similarity between representations within a model mainly depends on the depth and its architecture, e.g., repeated blocks and residual connections. GBS also reveals that models of different initialization learn different representations, especially in the deep layers. We compared the similarity of representations when the same batch of adversarial samples migrated to different models, and found that the same adversarial samples activated different features in different models. The similarity between representations of untargeted adversarial attacks is higher than that of targeted attacks.
6. Future Works

In the future, we will explore the application of GBS in other deep learning techniques, such as model quantification and adversarial training. We will also apply it to other domains, like natural language processing.

References

[1] N. Akhtar and A. Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. IEEE Access, 6:14410–14430, 2018.
[2] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PloS one, 10(7):e0130140, 2015.
[3] S. Boccaletti, G. Bianconi, R. Criado, C. I. Del Genio, J. Gómez-Gardenes, M. Romance, I. Sendina-Nadal, Z. Wang, and M. Zanin. The structure and dynamics of multilayer networks. Physics reports, 544(1):1–122, 2014.
[4] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. arXiv preprint arXiv:2005.14165, 2020.
[5] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In IEEE Symposium on Security and Privacy, pages 39–57, 2017.
[6] C. Cortes, M. Mohri, and A. Rostamizadeh. Algorithms for learning kernels based on centered alignment. The J. of Machine Learning Research, 13(1):795–828, 2012.
[7] A. Das and P. Rad. Opportunities and challenges in explainable artificial intelligence (xai): A survey. arXiv preprint arXiv:2006.11371, 2020.
[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In CVPR, pages 248–255, 2009.
[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.
[10] J. Frankle and M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. arXiv preprint arXiv:1803.03465, 2018.
[11] Y. Ge, Y. Xiao, Z. Xu, M. Zheng, S. Karanam, T. Chen, L. Itti, and Z. Wu. A peek into the reasoning of neural networks: Interpreting with structural visual concepts. In CVPR, pages 2195–2204, 2021.
[12] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Proc. of the Int. Conf. on Artificial Intelligence and Statistics, pages 249–256, 2010.
[13] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572, 2014.
[14] Y. Goyal, A. Feder, U. Shalit, and B. Kim. Explaining classifiers with causal concept effect (cace). arXiv preprint arXiv:1907.07165, 2019.
[15] Y. Guo, Q. Li, and H. Chen. Backpropagating linearly improves transferability of adversarial examples. arXiv preprint arXiv:2012.03528, 2020.
[16] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. Neural Computation, 16(12):2639–2664, 2004.
[17] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In ICCV, pages 1026–1034, 2015.
[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, pages 770–778, 2016.
[19] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry. Adversarial examples are not bugs, they are features. In Advances in Neural Information Processing Systems, pages 125–136, 2019.
[20] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In ICML, pages 448–456. PMLR, 2015.
[21] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton. Similarity of neural network representations revisited. In ICML, pages 3519–3529. PMLR, 2019.
[22] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.
[23] A. Kurakin, I. Goodfellow, S. Bengio, et al. Adversarial examples in the physical world, 2016.
[24] J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep neural networks as gaussian processes. arXiv preprint arXiv:1711.00165, 2017.
[25] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083, 2017.
[26] G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. Digital Signal Processing, 73:1–15, 2018.
[27] A. S. Morcos, M. Raghu, and S. Bengio. Insights on representational similarity in neural networks with canonical correlation. arXiv preprint arXiv:1806.05759, 2018.
[28] T. Nguyen, M. Raghu, and S. Kornblith. Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth. arXiv preprint arXiv:2010.15327, 2020.
[29] A. E. Orhan and X. Pitkow. Skip connections eliminate singularities. arXiv preprint arXiv:1701.09175, 2017.
[30] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. arXiv preprint arXiv:1706.05806, 2017.
[31] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy. Do vision transformers see like convolutional neural networks? arXiv preprint arXiv:2108.08810, 2021.
[32] V. Ramanujan, M. Wortsman, A. Kembhavi, A. Farhadi, and M. Rastegari. What’s hidden in a randomly weighted neural network? In CVPR, pages 11893–11902, 2020.
[33] R. Ranftl, A. Bochkovskiy, and V. Koltun. Vision transformers for dense prediction. In CVPR, pages 12179–12188, 2021.
[34] 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 ICCV, pages 618–626, 2017.

[35] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034, 2013.

[36] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[37] J. Song, Y. Chen, J. Ye, X. Wang, C. Shen, F. Mao, and M. Song. Depara: Deep attribution graph for deep knowledge transferability. In CVPR, pages 3922–3930, 2020.

[38] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. arXiv preprint arXiv:1412.6806, 2014.

[39] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199, 2013.

[40] S. Tang, W. J. Maddox, C. Dickens, T. Diethe, and A. Damianou. Similarity of neural networks with gradients. arXiv preprint arXiv:2003.11498, 2020.

[41] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. arXiv preprint arXiv:2105.01601, 2021.

[42] Y. Wang, H. Su, B. Zhang, and X. Hu. Interpret neural networks by identifying critical data routing paths. In CVPR, pages 8906–8914, 2018.

[43] Y. Wang, H. Su, B. Zhang, and X. Hu. Learning reliable visual saliency for model explanations. TMM, 22(7):1796–1807, 2019.

[44] J. You, J. Leskovec, K. He, and S. Xie. Graph structure of neural networks. In ICML, pages 10881–10891. PMLR, 2020.

[45] H. Yuan, L. Cai, X. Hu, J. Wang, and S. Ji. Interpreting image classifiers by generating discrete masks. PAMI, 2020.

[46] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In ECCV, pages 818–833. Springer, 2014.

[47] C. Zhang, P. Benz, T. Imtiaz, and I. S. Kweon. Understanding adversarial examples from the mutual influence of images and perturbations. In CVPR, pages 14521–14530, 2020.

[48] C. Zhang, A. Liu, X. Liu, Y. Xu, H. Yu, Y. Ma, and T. Li. Interpreting and improving adversarial robustness of deep neural networks with neuron sensitivity. IEEE TIP, 30:1291–1304, 2020.

[49] R. J. Zhang and Y. Y. Fred. Measuring similarity for clarifying layer difference in multiplex ad hoc duplex information networks. J. of Informetrics, 14(1):100987, 2020.

[50] Q. Zhou and L. Leydesdorff. The normalization of occurrence and co-occurrence matrices in bibliometrics using cosine similarities and chi-square coefficients. J. of the Association for Information Science and Technology, 67(11):2805–2814, 2016.

[51] S. Zhou, Y. Wang, D. Chen, J. Chen, X. Wang, C. Wang, and J. Bu. Distilling holistic knowledge with graph neural networks. In CVPR, pages 10387–10396, 2021.