# Subnetworks and Superposition

**Vignesh Pandiarajan**
Brown University
vignesh_pandiarajan@brown.edu

**Suraj Anand**
Brown University
suraj_anand@brown.edu

**Noah Foster**
Brown University
noah_foster@brown.edu

## Abstract

Modern neural networks are excellent at a variety of tasks, ranging from vision to text generation, but recently there has been a push for interpretability of these deep neural nets, birthing a new field called Mechanistic Interpretability. Mechanistic Interpretability seeks to reverse engineer neural networks, but due to the scale and complex computations of neural networks, phenomena such as superposition obscure what is actually happening. In an effort to improve transparency, this paper finds polysemantic neurons that demonstrate superposition, where a single neuron is used for identifying multiple concepts / features. We cluster high activating images for specific filters to verify that neurons are polysemantic then proceed to isolate the corresponding subnetworks. This paper uses continuous sparsification to isolate subnetworks, in contrast to other scoring metrics that yield much less sparse networks. This work adds to the growing body of work on interpretability, but specifically is one of the first works on causal dissection of neurons.

## 1 Introduction

Recently, a considerable body of research has delved into deciphering the mechanisms which Deep Neural Networks (DNNs) employ to approximate complex functions. Much of this research has revolved around reductionist analyses, focused on characterizing the specific weights, activations, and circuits through correlative and causal analyses [2, 3, 9, 13, 17, 23]. These works are part of a broader effort across deep learning to understand and manipulate the latent features learned by DNNs, resulting in techniques including linear probing, causal meditation analysis, and path patching. [1, 22, 23]. While this past research has given the deep learning community far better intuition on the probable mechanisms by which DNNs learn, the precise function of most specific neurons is still unknown.

The reason for this ambiguity is the effect of *superposition*. While some neurons map exactly to interpretable features, most do not; instead, specific neurons often map to two or more features, clouding interpretablility. This arises when there are more features than dimensions represented within the latent vector space of a specific hidden layer [4]. Each neuron may be thought of as an axis-aligned orthogonal direction in the hidden layer latent space. When there are more features than units in a hidden layer, the internal machinery of a DNN cannot cleanly represent each feature by an orthogonal direction. Thus, some proportion of neurons become *polysemantic* (i.e. represent multiple features). These polysemantic neurons are very difficult to interpret as they may excite or inhibit various features. However, prior art suggests that the activating effects polysemantic neurons may be separated into semantically similar groups [6, 16].

We hypothesize that within a particular neuron, semantically different groups arise via statistical independent processes. As a result, we should be able to find a *subnetwork* (also referred to as a

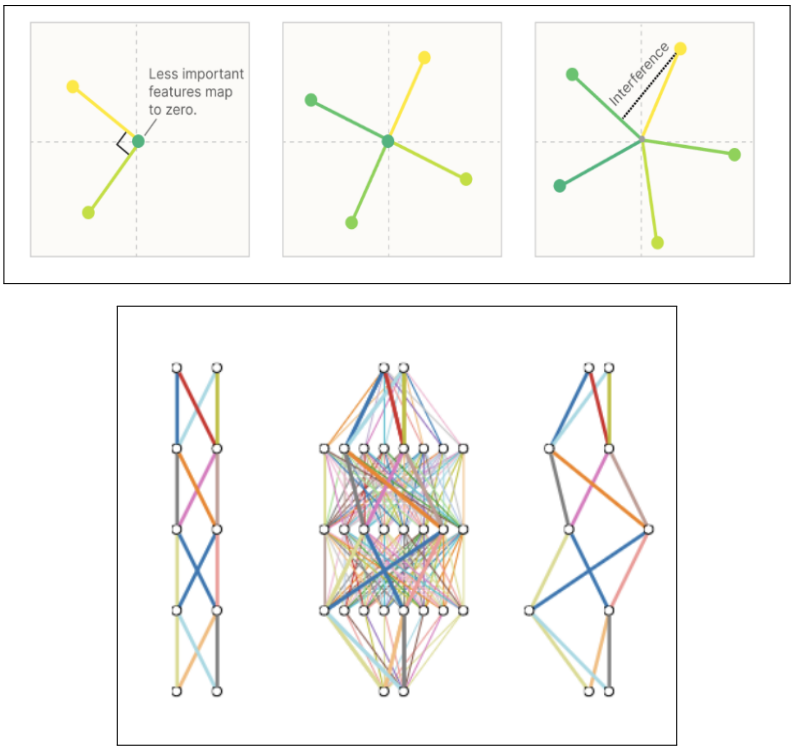


Figure 1: (**Top**) The Effect of Superposition [4], (**Bottom**) Subnetwork Extraction Depiction

circuit in some contexts) that highly activates only one semantically similar group in a polysemantic neuron. A subnetwork of a neural network is a sparse subset of neurons that isolates some particular task in the neuron network. A common perspective views DNNs as complex feature engineering machines with a final regression or classification head. Through this lens, subnetworks isolate specific feature computation routes in the larger logit generating graph. Past research has shown that sparse subnetworks may effectively isolate specific tasks in multi-task models and specific data-processing techniques [12, 19]. We compare the efficacy of heuristic scoring methods and L0 regularization techniques for subnetwork creation to isolate sparse monosemantic routes from originally polysemantic neurons. This is an invaluable exploratory tool to causally decipher the inner workings of DNNs at a neuron-level analysis.

Our main contributions are:

- We identify a number of polysemantic neurons in Sparse AlexNet convolutional layers six, eight, and ten through Hierarchical Based Density Clustering. We qualitatively characterize the semantic groups that generate excitatory responses in these neurons.
- We convert each polysemantic neuron to a set of subnetworks for which the neuron is monosemantic. We compare subnetwork creation using score-based heuristics and continuous $L_0$ regularization.
- We develop evidence to confirm that semantically different groups arise via statistical independent processes. This evidence is important for a broader discussion of how neural networks generate latent spaces.

## 2 Related Works

We build on recent research in mechanistic interpretability and network sparsification.

Elhage et al. demonstrates that superposition is observable in multiple toy models. Elhage also shows that (1) computation is possible in superposition, and (2) network representations are composed of many independently interpretable features represented by directions [4]. This makes the relationship

between superposition and polysemanticity clear. Since a polysemantic neuron recognizes multiple different features or concepts, computation would be in superposition if features are unrelated.

The field neural network sparsification dates back to the late 1980s with Mozer & Smolensky [15]. There has been significant work in pruning neural networks to increase weight efficiency and reduce compute in application [7, 8, 14, 24]. Recently, there has been much research on extracting subnetworks, which may be thought of as pruning a network, but for an objective function that differs than the original objective function [12, 18, 20]. We utilize past heuristic scoring [10, 14, 15] and $L_0$ regularization techniques [20, 21] for pruning. This research was inspired by the "Lottery Ticket Hypothesis," which suggests combinatorial optimization of randomly initialized networks can approximate useful functions [5].

Hamblin et al. serves as the key motivation for this paper, as they provide a strong framework for pruning to find networks that activate polysemantic neurons in a monosemantic manner[6]. They create a strong pipeline to (1) identify polysemantic neurons, and (2) prune circuits using saliency based criteria. We differentiate ourselves from this research by utilizing subnetwork discovery mechanisms based on optimization with $L_0$ regularization, rather than simple scoring heuristics. Moreover, we provide important combinatorial analysis on the subnetworks that comprise polysemantic neurons.

## 3 Methods

### 3.1 Finding Polysemantic Neurons

We attempt to find polysemantic filters in CNNs, meaning they activate for images of multiple distinct and separate categories. These filters raise the question of what computations are responsible for their selective activations across various categories. Take for example a filter that activates strongly on both porcupines and dogs. There are two possible situations now, that the filter responds to some commonality between both porcupines and dogs or that the filter is encoding separately for both porcupines and dogs. If the filter is responding to some similarity between porcupines and dogs, that means that it would be very difficult to separate the activations of the two different classes. However, if the filter was encoding them separately and they were put together just for the purpose of network compression, then it would suggest different computations are leading to these high activations [6]. First, to find these polysemantic filters, we pass all images in ImageNet through the network,

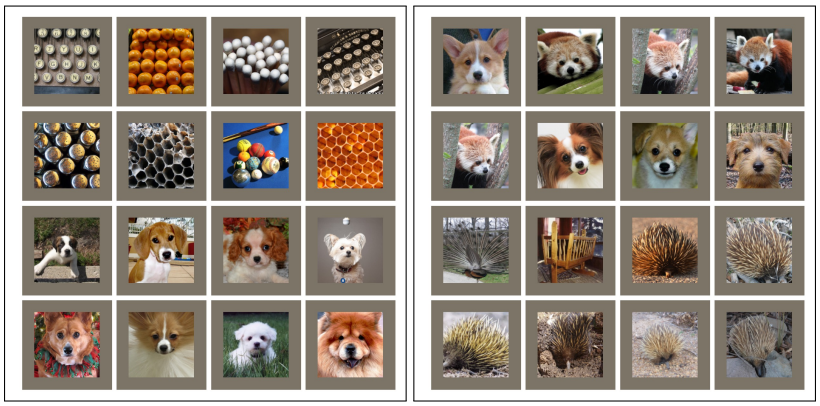

Figure 2: Sample Clusters for Polysemantic Filters
**(Left)** AlexNet Convolutional Layer 10, Unit 15
**(Right)** AlexNet Convolutional Layer 10, Unit 167

recording activations for a selected filter in a selected layer. Using receptive fields, we only record the activation of the filter when it is at the middle of the image. Hence, these activations are all scalars, so we know which images activate the filter the most. Now that we have the images that produce the highest activations on the filter, we now want to see if the filter is truly polysemantic. To do this, we record the center activations of all filters at the specified layer. We then apply HDBSCAN, a clustering algorithm to this high dimensional data. As mentioned earlier, if we see multiple clusters then we hypothesize polysemanticity exists in the selected filter.

#### 3.1.1 Receptive Fields

Receptive fields are the regions that can affect the activation of a filter from the immediately previous layer. For example, if our filter is a $[5 \times 5]$ filter at layer 1, for an input size of $[32 \times 32 \times 3]$, then the receptive field will be of size $[5 \times 5 \times 3]$. When computing an activation, the receptive fields is essentially the salient region of the previous layer. Effective receptive fields are similar, but are the salient region of the original image rather than the previous layer. In CNNs the ERF grows from layer to layer because more of the image affects the activation [11]. Taking this into account, images were cropped to the ERF of the center application of the filter that we wished to experiment on. This is how we were able to extract scalar activations that still represented the entire image.

#### 3.1.2 Subnetwork Pruning

After we find these polysemantic neurons and cluster these images, we aim to show these neurons are truly polysemantic. Using pruning we can find subnetworks that correspond to one of the polysemantic meanings. We want to preserve the network's ability to activate on one of the two clusters while reducing the number of overall parameters. In order to do this we use several scoring methods and a masking method from Hamblin et al. [6]. If the found subnetworks show significantly different activations for images other than their respective cluster, then we have found the part of the network that corresponds to the cluster in question.

### 3.2 Pruning Scoring Methods

We use Hamblin et al.'s feature-wise saliency criteria to develop a baseline for subnetwork extraction performance. [6]. These methods approximate $|\Delta f|$ where $f$ is the objective function for which the subnetwork is evaluated. Specifically, we test **ActGrad, SNIP**, and **FORCE**. We evaluate the viability of each of the below methods on extracting subnetworks which convert each polysemantic neuron to monosemantic in semantically similar images.

#### 3.2.1 ActGrad

This scoring procedure was designed for filter pruning. It computes score with respect to a filter's activation map [14].

$$\boldsymbol{S_a}(\theta_j; f, \boldsymbol{x}) = \frac{1}{H_{out}W_{out}} \sum_{w=1}^{W_{out}} \sum_{h=1}^{H_{out}} |a_{h,w} \frac{\partial f(\boldsymbol{x})}{\partial a_{h,w}}|$$

#### 3.2.2 SNIP

Originally introduced in Mozer & Smolensky, this criterion computes score with respect to the weights in a kernel rather than activations in order approximate the change in an objective function [15].

$$\boldsymbol{S_s}(\theta_j; f, \boldsymbol{x}) = \frac{1}{K_w K_h} \sum_{w=1}^{K_w} \sum_{h=1}^{K_h} |\boldsymbol{w}_{h,w} \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{w}_{h,w}}|$$

#### 3.2.3 FORCE

This scoring method iteratively computes **SNIP** on a sparser and sparse subnetwork in the hope of approximating $\theta$'s importance in the subnetwork rather than the original dense network. The intuition behind this is that removing a particular $\theta$ may have a completely different effect when other parameters are being pruned than when it is complete [6, 10].

### 3.3 Continuous Sparsification

All of the above methods are heuristic saliency-based methods. We want to search for or learn the subnetwork in a way that is free of assumptions about the target sparsities. Finding a subnetwork that contributes a feature to a polysemantic neuron is equivalent to learning a mask over the network specifying membership to the subnetwork. Learning the binary values of the mask is a search over a

discrete space too large to brute force search. In order to learn the mask, we apply the Continuous Sparsification technique introduced by Savarese et al. [20]. For every parameter $p$ that we would like to learn a mask for, we create a parameter $m$ so that $\sigma_t(m)$ is the mask for $p$ where $\sigma_t(x) = \frac{1}{1+e^{-tx}}$ is the Sigmoid function. We will freeze the original weights of the network so that we are only learning this mask. After each epoch, we raise the temperature $t$, pushing $\sigma_t(x)$ closer to 1 or 0 and further from 0.5. Thus the surface we are optimizing over becomes closer and closer to the step function we want. We follow an exponentially annealing schedule for temperature increase. We let temperature range from 1 to 200 by scaling the temperature by $200^{1/(n-1)}$ when training for $n$ epochs.

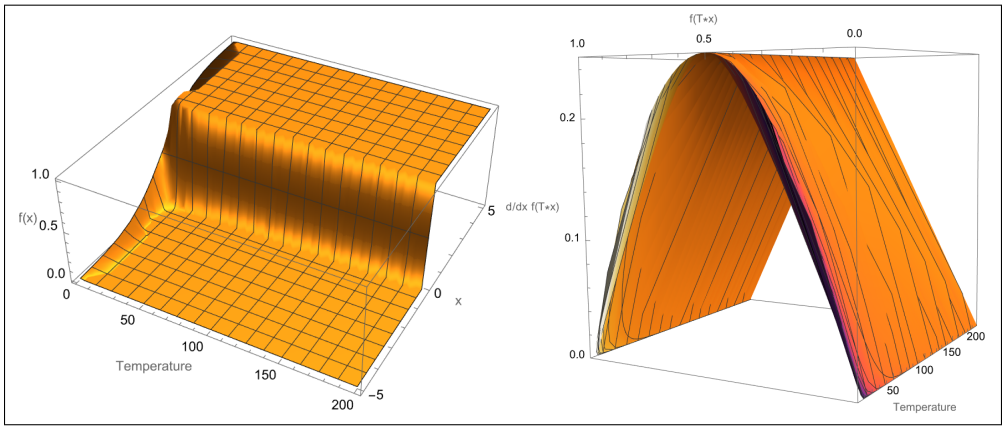


Figure 3: **(Left)** Sigmoid Function with Temperature between 1 and 200
**(Right)** $\frac{\mathrm{d}}{\mathrm{d}x}\sigma_t(x)$ Plotted Against $\sigma_t(x)$ with Temperature between 1 and 200

For any $t$, optimizing this mask is a differentiable. We apply stochastic gradient descent with momentum and a loss function defined below to find a mask.

We are interested in finding the weights in the network that contribute most to the high activation of our neuron of interest for one cluster. Because high activation for our cluster only has meaning relative to the activation of everything else, we will use Binary Cross Entropy Loss against an indicator function of membership to the cluster of interest to emulate emulate a contrastive loss. In order to ensure that we are learning a subnetwork, we also include a term to encourage sparsity in our loss. This is complex because we do not want to penalize the mask values close to 1, but we want to encourage those in the middle to descend to 0. Thus the regularization we want is $L_0$ loss, which for a model with parameters $\mathcal{P}$, is defined to be

$$L_0(\mathcal{P}) := \sum_{p \in \mathcal{P}} \mathbb{1}_{p \neq 0}$$

We approximate with the sum of the values of the mask. This is a very effective approximation as when $\sigma_t(x)$ is close to 0 or 1, the derivative is small, but $\frac{\mathrm{d}}{\mathrm{d}x}\sigma_t(x)$ is maximized when $\sigma(x) = 0.5$. This property also varies very little with temperature. As noted in Savarese et al. [20], the Continuous Sparsification method is powered by this L0 regularization. This follows from the fact that masking can be represented as an L0 problem. This is confirmed by our results showing that this masking objecting is extremely effective.

### 3.4 Neuron Visualization

In order to generate visualizations, we used Lucent, a library for neural network interpretability and visualization. We used Lucent to visualize which features activate a selected filter. This is done through first generating a random image, then performing gradient descent on the image itself to see what image will generate the strongest activation from the filter in question. If our approach is successful, we would see multiple concepts in the visualization of the filter before network pruning and then see singular concepts afterwards.

# 4 Results

## 4.1 Subnetwork Metrics

Heuristic scoring methods for identification of subnetworks require target sparsities. These scoring methods fail to be effective below 10% target sparsity. Our method does not as we chose whether to include each particular candidate neuron in the subnetwork not by how it's salience compares to other candidate neurons, but directly with a loss function on the activations of the downstream neuron of interest. This brings great interest to the question what sparsity will our method yield. We find that we consistently find subnetworks with 3%-7% sparsity. This shows that our technique more stably identifies the core subnetwork that contributes a given feature. Furthermore, we assess how well monosemantic subnetworks were identified by qualitatively observing how the distance between the activation distibutions were pushed apart.

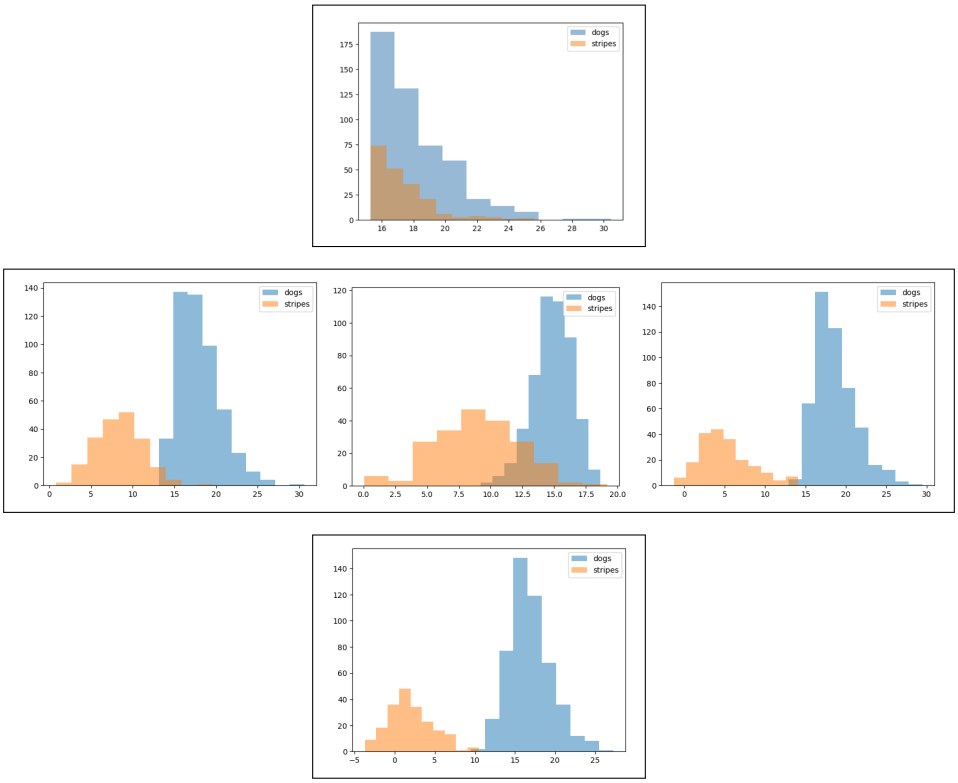


Figure 4: AlexNet Layer 10 Unit 255 Activations
**(Top)** Top 1000 Dog and Stripe Activations
**(Middle)** ActGrad, SNIP, and FORCE Dog Subnetwork Activations
**(Bottom)** Continuous Sparsification Dog Subnetwork Activations

Above are selected results for Convolutional Layer 10, Unit 255, which has two many semantic clusters corresponding to dogs and stripes. As seen in the above graphs, these two activations are initially overlapping but can be separated into two distributions using both Heuristic and $L_0$ regularization methods. As seen in the above graphs, FORCE and $L_0$ regularization perform the best as the two cluster distributions are separated the most. This trend was also seen in Units 167 and 15 of Convolutional Layer 10 and Unit 25 of Convolutional Layer 8.

## 4.2 Visualizations

Visualization analysis necessarily has to be qualitative. In clustering Convolutional Layer 10, Unit 255 which corresponds to both stripes and dogs, the visualizations appear to show that the subnetwork trained to boost the activations of stripes (the stripe subnetwork) show more alternation. This would cause increased activation at images of stripes. However there are many artifacts in these images that

less convincingly belong to either class. Most generated images have large white spots. Two possible causes are the the limited size of the clusters or the sparsity of the networks. Further work would ablate both of these factors. The images below are for the same neuron and clusters, but are produced with ActGrad, which produces much less sparse subnetworks.

Figure 5: Backpropagation generated images for Layer 10, Unit 255
**(Left)** Original
**(Center)** Subnetwork for Stripes
**(Right)** Subnetwork for Dogs

## 5 Discussion

In this research, we explore the hypothesis that *superposition* in DNNs arises through statistically independence processes. We find strong supporting evidence in that we can identify distinct, sparse subnetworks that contribute to each semantic function of a polysemantic neuron. We show that approximating $L_0$ regularization through temperature-scaled sigmoided masks outperforms objective function preservation scoring heuristics from Hamblin et al. However, we find that the $L_0$ regularization procedure shown in continuous sparsification sometimes meaningfully alters activation behavior for examples that are not included in the continuous sparsification dataset. Finally, we visualize original and subnetwork neurons to qualitatively assess polysemanticity dissection quality.

In the broader context of mechanistic interpretability, we believe that this research represents an important method for causal dissection of neurons. While methods like path-patching and neuron ablation show the effects of altering or zeroing activations on resultant logits, this method represents a first step into direct analysis of neuron function. Future research should (1) analyze the discovered monosemantic subnetworks, (2) identify deterministic behavior in neural polysemanticity among different neural networks or random initializations, and (3) develop methods to dissect more complex polysemantic kernels. We would also enjoy seeing future research that replicates results in a natural language domain or using transformers.

## 6 Acknowledgements

We would like to acknowledge Chen Sun for his direction over the course of this project and this class as well as Michael Lepori for a number of discussions that were critical to the generation of the hypothesis and methods. This project was conducted for the *Advanced Topics in Deep Learning* course at Brown University.

## References

[1] Guillaume Alain and Yoshua Bengio. *Understanding intermediate layers using linear classifier probes*. 2018. arXiv: 1610.01644 [stat.ML].

[2] Nick Cammarata et al. "Thread: Circuits". In: *Distill* (2020). https://distill.pub/2020/circuits. DOI: 10.23915/distill.00024.

[3] Nelson Elhage et al. "A Mathematical Framework for Transformer Circuits". In: *Transformer Circuits Thread* (2021). https://transformer-circuits.pub/2021/framework/index.html.

[4] Nelson Elhage et al. "Toy Models of Superposition". In: *Transformer Circuits Thread* (2022). https://transformer-circuits.pub/2022/toy$_m odel/index.html$.

[5] Jonathan Frankle and Michael Carbin. *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks*. 2019. arXiv: 1803.03635 [cs.LG].

[6] Chris Hamblin, Talia Konkle, and George Alvarez. *Pruning for Feature-Preserving Circuits in CNNs*. 2023. arXiv: 2206.01627 [cs.CV].

[7] Song Han, Huizi Mao, and William J. Dally. *Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding*. 2016. arXiv: 1510.00149 [cs.CV].

[8] Song Han et al. *Learning both Weights and Connections for Efficient Neural Networks*. 2015. arXiv: 1506.02626 [cs.NE].

[9] Boris Hanin and David Rolnick. *Deep ReLU Networks Have Surprisingly Few Activation Patterns*. 2019. arXiv: 1906.00904 [stat.ML].

[10] Pau de Jorge et al. *Progressive Skeletonization: Trimming more fat from a network at initialization*. 2021. arXiv: 2006.09081 [cs.CV].

[11] Hung Le and Ali Borji. *What are the Receptive, Effective Receptive, and Projective Fields of Neurons in Convolutional Neural Networks?* 2018. arXiv: 1705.07049 [cs.CV].

[12] Michael A. Lepori, Thomas Serre, and Ellie Pavlick. *Break It Down: Evidence for Structural Compositionality in Neural Networks*. 2023. arXiv: 2301.10884 [cs.CL].

[13] Lu Lu. "Dying ReLU and Initialization: Theory and Numerical Examples". In: *Communications in Computational Physics* 28.5 (June 2020), pp. 1671–1706. DOI: 10.4208/cicp.oa-2020-0165. URL: https://doi.org/10.4208%2Fcicp.oa-2020-0165.

[14] Pavlo Molchanov et al. *Pruning Convolutional Neural Networks for Resource Efficient Inference*. 2017. arXiv: 1611.06440 [cs.LG].

[15] Michael C Mozer and Paul Smolensky. "Skeletonization: A Technique for Trimming the Fat from a Network via Relevance Assessment". In: *Advances in Neural Information Processing Systems*. Ed. by D. Touretzky. Vol. 1. Morgan-Kaufmann, 1988. URL: https://proceedings.neurips.cc/paper_files/paper/1988/file/07e1cd7dca89a1678042477183b7ac3f-Paper.pdf.

[16] Jesse Mu and Jacob Andreas. "Compositional Explanations of Neurons". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 17153–17163. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/c74956ffb38ba48ed6ce977af6727275-Paper.pdf.

[17] Neel Nanda et al. *Progress measures for grokking via mechanistic interpretability*. 2023. arXiv: 2301.05217 [cs.LG].

[18] Chris Olah et al. "Zoom In: An Introduction to Circuits". In: *Distill* (2020). https://distill.pub/2020/circuits/zoom-in. DOI: 10.23915/distill.00024.001.

[19] Jacob Renn et al. *The Multiple Subnetwork Hypothesis: Enabling Multidomain Learning by Isolating Task-Specific Subnetworks in Feedforward Neural Networks*. 2022. arXiv: 2207.08821 [cs.LG].

[20] Pedro Savarese, Hugo Silva, and Michael Maire. *Winning the Lottery with Continuous Sparsification*. 2021. arXiv: 1912.04427 [cs.LG].

[21] Suraj Srinivas, Akshayvarun Subramanya, and R. Venkatesh Babu. *Training Sparse Neural Networks*. 2016. arXiv: 1611.06694 [cs.CV].

[22] Jesse Vig et al. "Investigating Gender Bias in Language Models Using Causal Mediation Analysis". In: *Neural Information Processing Systems*. 2020.

[23] Kevin Wang et al. *Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 small*. 2022. arXiv: 2211.00593 [cs.LG].

[24] Ruichi Yu et al. *NISP: Pruning Networks using Neuron Importance Score Propagation*. 2018. arXiv: 1711.05908 [cs.CV].