Uncovering the Internal Representations of Neural Networks with Regression Analysis

Hiroshi Nonaka

May 12, 2025

Abstact

With the rise of AI technologies in different social sectors, understanding the internal states of deep neural networks is becoming a critical research field that tackles the black box problem. One possible approach is probing, a regression analysis that takes internal activations as independent variables and regresses target information. This method quantifies how informative the representations are in relation to the dependent variables. I implemented probing for a simple three-layer neural network for binary image classifications with three regression models: linear probability models, logistic regression, and neural networks. The results revealed that all three layers possess equally rich information for the binary image labels. Through further statistical analysis, I discovered that the representation vectors from deeper layers are more multicollinear. By applying Principal Component Analysis (PCA) to mitigate multicollinearity, I also found out that the vectors essentially acquired an equally useful representation. Heteroskedasticity tests showed that the data was heteroskedastic at a high significance level, and the training steps did not affect it.

1 Introduction

1.1 Overview

Neural networks are essential building components of recent AI models such as ChatGPT [1]. Although neural network predictions often outperform linear regression models because of their flexible representation capacity, their internal states are hard to interpret compared to linear models. For instance, the common multivariate linear regression can be written in the following form:

$$\hat{Y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$$

where $\hat{Y}$ is an estimated value of a dependent variable Y, and β_i are the coefficients of independent random variables X_i. This is easy to interpret. In contrast, neural networks with m layers are often represented as:

$$\hat{Y} = (T_l \circ f_{l-1} \circ T_{l-1} \circ \dots f_1 \circ T_1)(\vec{X}) \tag{1}$$

where $\circ$ is the function composition operation, $\vec{X} = [X_1, X_2, \dots, X_n]$, T_i is a transformation such that $T_i(\vec{V}) = A_i\vec{V} + b_i$ for some input vector $\vec{V}$, matrix A_i and bias vector b_i, and f is some function (mostly non-linear) that slightly modify outputs from T_i. Repetitively applying $f \circ T$ gives rise to the non-linearity of neural network functions. We *train* neural networks so they learn patterns in a dataset through multiple training steps. During the training, the error between the output $\hat{Y}$ and the true value Y is calculated. This error function is called a *loss function*, and neural networks update the values of their matrices A_i and bias vectors b_i by taking the gradient of the loss function. This is how training optimizes neural networks to *fit* to the given dataset.

1.2 Motivation

Due to the multiple transformations and non-linear functions within neural networks, it becomes hard to interpret what the matrix and bias vector in each transformation represent compared to simple linear regression models, in which the coefficients denote how much each variable contributes to the estimated point $\hat{Y}$. The uninterpretability of neural networks is named the *black box problem* and has stimulated various research endeavors, including *explainable AI (XAI)* [6]. Given that AI systems are implemented in many different sectors today, understanding *how AI thinks* is one of the critical themes of ongoing research. I aim to tackle this problem through a popular approach called probing [5]. My research questions are mentioned in Section 4.

1.3 Approach

Probing applies regression analysis to measure how rich information an internal vector contains in relation to target data. Let an *activation* $\vec{Y_i} = [Y_{i,1}, Y_{i,2}, \dots, Y_{i,m}]$ represent the output vector of the i-th transformation T_i in Equation 1: $\vec{Y_i} = T_i(f_{i-1}(\vec{Y_{i-1}}))$. Note that the first activation is produced from the observation vector: $\vec{Y_1} = T_1(\vec{X})$. In prob-

ing, regression models P, namely *probes*, regress some target information using the vector $\vec{Y_i} = [Y_{i,1}, Y_{i,2}, \ldots, Y_{i,m}]$ as m independent variables. If P produces good predictions, this indicates that $\vec{Y_i}$ are good independent variables, or more specifically, contain rich information about the target.

2 Notations

Here, I outline notations that frequently appear in this paper. T_i indicates the i-th transformation layer in a neural network. f_i represents a function applied to the output from T_i. Most models uses a single non-linear function throughout their pipeline, so I occasionally write f to denote the unique function. $\vec{X} = [X_1, X_2, ..., X_n]$ is an n-dimensional observation and an input to a neural network. An activation $\vec{Y_i} = [Y_{i,1}, Y_{i,2}, \ldots, Y_{i,m}]$ is an m-dimensional vector from the i-th layer in the neural network. Therefore, each observation $\vec{X}$ has corresponding activations $\vec{Y_1}$, $\vec{Y_2}$ through $\vec{Y_l}$ for each transformation in Equation 1. In my paper, P is a probe/function/regression model that takes activations and maps them to a probability p: $p = P(\vec{Y_i})$. One example of P is logistic regression.

3 Literature Review

In this section, I aim to introduce some prior works on probing and simultaneously deepen the audience's understanding of its method through concrete examples.

3.1 Machine Learning in Regression Analysis

Before discussing probing techniques, it is worth noting that machine learning algorithms are closely tied to regression analysis in many fields. Decision trees [10] and their families, such as random forest and gradient boosting, are popular regression algorithms for their accuracy and interpretability. Ridge and Lasso regressions [13], regularized linear regression models, are also important linear models in machine learning. These machine learning models are applied in many different areas, such as political sciences [7], chemistry [4], and economics [15].

3.2 Probing Neural Networks

Gurnee and Tegmark [8] applied Ridge regression and neural nets to measure the spatiotemporal representations within large language models (LLMs). LLMs are neural-network-based AI models that deal with language, such as ChatGPT. For spatial information, they gave LLMs prompts asking for the latitude and longitude of a specific place in the world, and for temporal information, the LLMs were asked about historic dates related to well-known books, figures, and et cetera. Activations produced by those stimuli were collected and used as independent variables to regress the latitudes and longitudes of the places or the associated years to the historic events. The researchers discovered that the linear model

can predict the dependent variables with high accuracy, concluding that the spatio-temporal representations within LLMs are linearly related to the target values.

Multiple research works have applied probing to investigate representations of neural networks in various modalities, like text and image. McCoy and Leshinskaya [12] applied regression analysis to investigate how an LLM understands subject-object relations. Specifically, they focused on the *has-color* relation; for instance, the subject is *bananas*, and the relation between the subject and object is *have the color of*, then the expected object is *yellow*. They gave prompt queries asking the color of a particular subject, collected activations from the fifth layer of the LLM ($\vec{Y_5}$), and regressed the final output $\vec{Y_l}$.
Alain and Bengio [3] introduced linear classifier probes as a tool to understand the information captured in intermediate layers of neural networks. They collected activations from each layer of their neural network for image classification. Results revealed that the error of probes decreased as they used activations from deeper (later) layers as independent variables. Rashtchian et al. [17] probed image encoders of foundation models (AI models processing multimodal data) to quantify how the encoders generalize image representations. They collected activation vectors by giving the same images, but with different visual effects, and discovered that the representations that the encoders obtain from these differentially edited images converge well. Ilharco et al. [9] focused on the multimodal relations between contextual text representations with corresponding visual features by regressing image features based on activations from text-only LLMs. Results indicated that language representations are strong signals for predicting visual features.
Akhondzadeh et al. [2] leveraged linear, Bayesian, and pairwise probing to investigate the relations between the representations of graph-based models with important chemical properties like functional groups and odors. Ngo and Kim [14] trained probes to align language representations in LLMs with sound representations from audio models.

4 Research Directions and Questions

Despite the considerable number of prior studies, few have focused on the statistical analysis, such as heteroskedasticity test and multicollinearity, of representations along with probing. In this work, I aim to (1) mildly replicate existing research on probing techniques and (2) apply statistical analysis to internal representations for further understanding than probing. The research questions in this work are (1) which activation outputs $\vec{Y_i}$ contain rich information, and (2) how the amount of training affects representation learning.

5 Data Collection

5.1 Training and Architecture

To collect activation vectors, I first trained my neural network with 2,000 images of cats and dogs from the CIFAR10 dataset [11]. The model classifies whether a given image is likely to be a cat or a dog. The training pipeline is as follows: (1) give the model an image and have

it predict probability p that the image is a cat; (2) calculate binary cross-entropy between p and its true label as an error score; (3) repeat step (1) and (2) for 1,600 training images; (4) calculate a validation score with 400 unseen data; and (5) repeat step (1) through (4) for 100 times with the same training and validation data. Each repetition of steps (1) through (4) is called an *epoch*.

The model consists of a *convolution* part and a neural net part. Convolution layers compress a $64 \times 64 \times 3$ (height, width, and channels) image into $13 \times 13 \times 16$. Note that the convolution layers are also trainable.
The neural network has three layers: T_1, T_2 and T_3. Each T_i maps $f(\vec{Y}_{i-1})$ or $\vec{X}$ to a vector of 60 dimensions. Therefore, $T_1 : \mathbb{R}^{2704=13\cdot13\cdot16} \to \mathbb{R}^{60}$, $T_2 : \mathbb{R}^{60} \to \mathbb{R}^{60}$, and $T_3 : \mathbb{R}^{60} \to \mathbb{R}^{60}$. A non-linear function f called *Rectified Linear Unit* follows T_1, T_2, and T_3. One final layer T_4 transfoms the 60-dimensional vector from the previous transformation T_3 to a scalar. The logistic function σ turns the scalar into probability p.
The overall model architecture is outlined below:

$$p = (\sigma \circ T_4 \circ f \circ T_3 \circ f \circ T_2 \circ f \circ T_1 \circ conv)(\vec{X})$$

where *conv* is the convolution part, and $\vec{X}$ is an image vector.

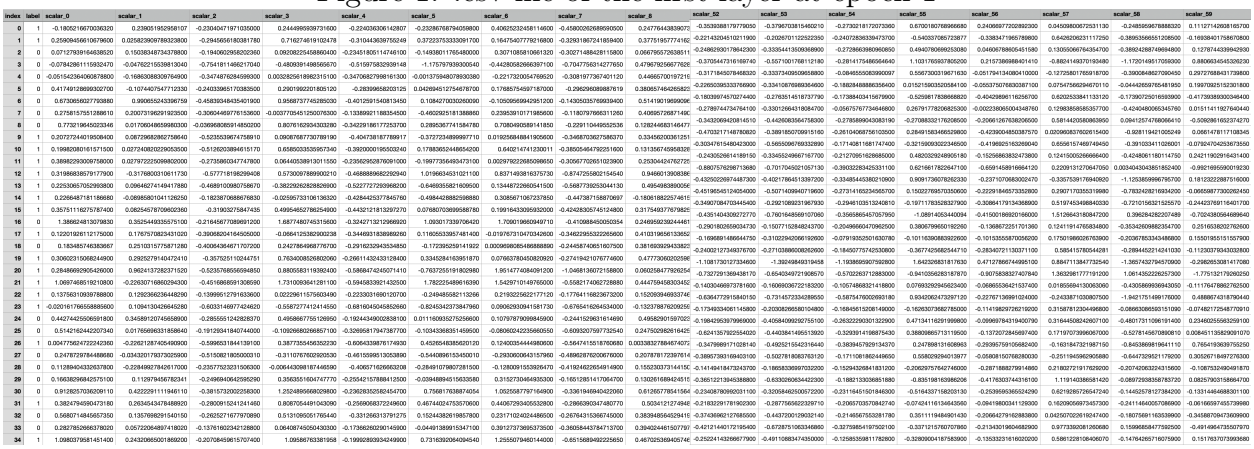
5.2 Data Requirement

After each *epoch* during training, I gave the same 2,000 images to the network and collected $\vec{Y_1}$, $\vec{Y_2}$, and $\vec{Y_3}$ for *each image*. Therefore, I collected $600,000 (= 2,000$ images $\times$ 3 layers $\times$ 100 epochs) activations in total. I restricted the output dimension of T_1, T_2, and T_3 to 60 in order for $\vec{Y_i}$ to have the same dimension size, which, later in probing, prevents the variability of regression performance due to the different number of independent variables. In short, the data I collected is cross-sectional data consisting of 300 .csv files (3 files for each of the three layers, multiplied by 100 epochs) with a table of 60 independent variables ($\vec{Y_i}$) and 2,000 observations for each file (refer to Figure 1 for details). I did not implement any control or instrument variables because of the nature of my research. The details of data preprocessing are explained in Section 6.2.

5.3 Assumptions and Diagnostics

For statistical analysis, I focused on two metrics, heteroskedasticity and multicollinearity. For heteroskedasticity, I ran the Breusch-Pagan test to calculate p-values for the null hypothesis of homoskedasticity. The left plot in Figure 2 shows that the entire data is heteroskedastic at a very high significance level (p-value $<<<$ 0.001). Only the first layer experiences rapid increases of p-value, but this trend is inconsistent.
VIF of the second and third layers are also extremely high. VIFs of the third layer activations converges around 3 as the epoch approaches the end of training. Therefore, the activations from the first layer are mildy multicollinear, and those of the second and third layers have high multicollinearity. However, I can observe the downward trend of VIF as the amount

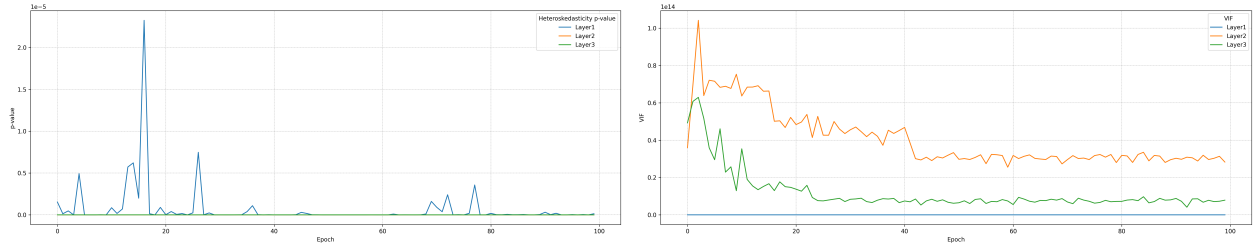
Figure 1: .csv file of the first layer at epoch 1



The table includes expected labels and vector values as columns. *scalar_j* denotes the j-th value ($Y_{i,j}$) in the vector ($\vec{Y_i} = [Y_{i,1}, \ldots, Y_{i,j}, \ldots, Y_{i,m}]$). Indexes on the leftmost column number different 2,000 activations.

of training increases and the mean VIF converges to certain value points (this trend is also observable in the third layer).

Figure 2: Heteroskedasticity and VIF



The left line plot is the p-values for the heteroskedasticity test, and the right graph shows the mean VIF within data from each layer and epoch. Note that the y-axis of the left graph is scaled by 1e-5 (1.0 in the y-axis is equal to 0.000001), and VIF is scaled by 1e14 = 100,000,000,000,000.

6 Methodological Framework

6.1 Probes

I prepared three regression models as probes. The first is linear probability models:

$$p = \beta_0 + \beta_1 Y_{i,1} + \beta_2 Y_{i,2} + \cdots + \beta_{60} Y_{i,60}$$

Although this model cannot capture the trend of cumulative Gaussian distribution functions, it serves as a good measure of the linearity of representations because it directly applies linear regression for probability p. In other words, if a linear probability model predicts probabilities with high accuracy, the independent variables $Y_{i,1}, Y_{i,2}, \ldots, Y_{i,60}$ have high linearity with

target labels. We clipped any $p > 1$ as 1 and $p < 0$ as 0.
The second model is the logistic regression model. Logistic regression captures the natural shape of the cumulative Gaussian distribution functions and is a popular model for binary classification.

$$p = \frac{1}{1 + exp(-(\beta_0 + \beta_1 Y_{i,1} + \beta_2 Y_{i,2} + \cdots + \beta_{60} Y_{i,60}))}$$

Finally, I introduce simple three-layer neural networks. Neural networks serve as a good indicator of non-linearity in activations. If the accuracy of linear probability models and logistic regression is low while the neural networks regress well, there is a non-linear relationship between the activations and true labels. I applied the three models to each of the 300 .csv files and recorded the accuracy. I split 2,000 observations into training and test datasets.

6.2 Data Preprocessing

Most values of my data range from 0.3 and -0.3, which resulted in the poor convergence of the regression models. Therefore, I standardized the values. Standardization is given by $x_{std} = \frac{x - \mu}{\sigma}$ where μ is the mean of the independent variable X, which sample x belongs to, and σ is the standard deviation of the variable. In the follow-up research mentioned later, I also performed principal component analysis (PCA) to reduce the dimensions of the observations from 60 to 10. PCA essentially removes unnecessary, sparse representations and compresses data to lower dimensions while retaining most of the information contained in the original data.

7 Results

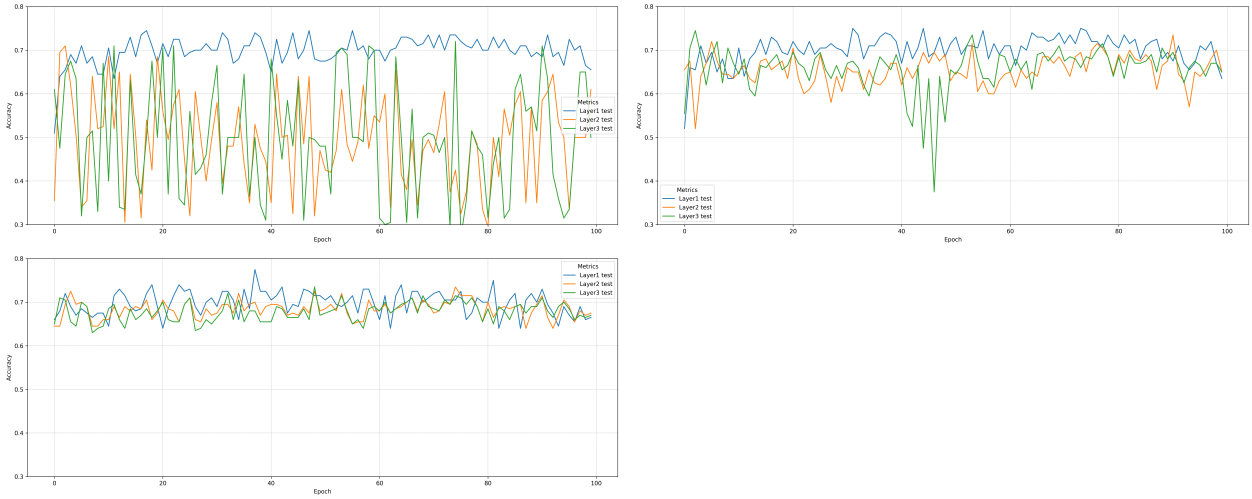
7.1 Probing Results

Figure 3 shows the results of probing. The linear probability models suffer from fluctuating accuracy. Logistic regression is more robust than linear probability models, but there still exists a large decline in accuracy. In contrast, neural networks regressed the true labels more stably. Moreover, the overall results show that the first layer contains more stable representations, while the activations from the second and third layers resulted in slightly lower or significantly lower accuracy in the logistic models and linear probability models. These graphs present counterintuitive results, as I initially speculated that representations would grow richer as the layer goes deeper, closer to the output layer. However, in Section 4, I identified the multicollinearity in the activation data and hypothesized that it degraded the accuracy of the probes.

7.2 Probing Results with PCA

Speculating that the high multicollinearity contributes to the low accuracy with the second and third layers, I performed PCA to reduce dimensionality. PCA mitigates multicollinearity by creating new variables that are linearly unrelated to each other [16]. Probing results

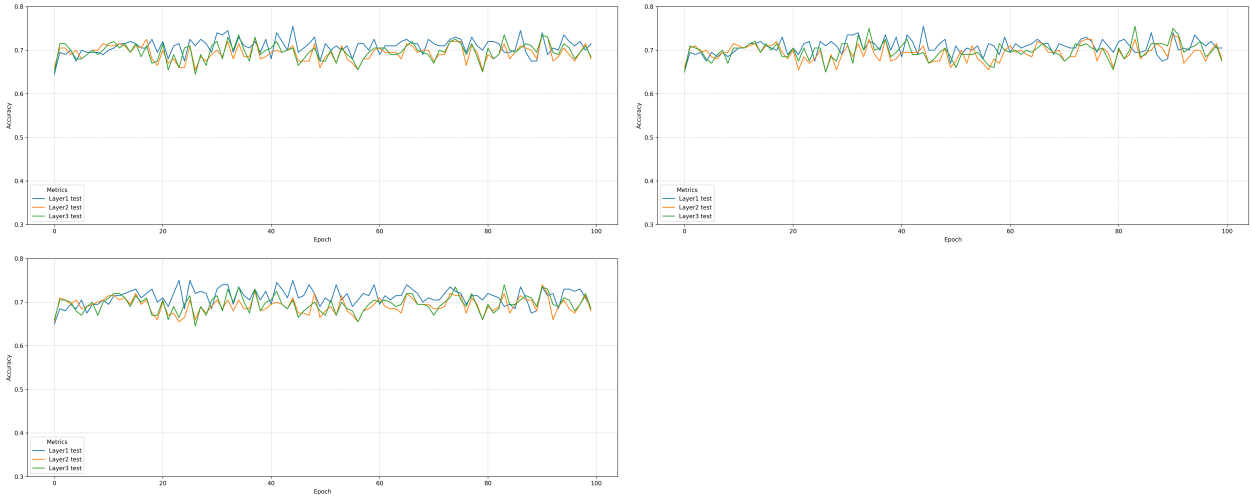
Figure 3: Accuracy of Probes on Test Data



The upper-left graph is the linear probability models, the upper-right is the logistic models, and the bottom-left is the neural networks. The blue lines represent the scores of probes on the activations from the first layer, the orange lines correspond to the second layer, and the green lines track that of the third layer.

after PCA are outlined in Figure 4. I can observe clear improvements in the accuracy scores across the three probes, especially in the linear probability models. Furthermore, the accuracy scores across different layers generally stay at almost the same level, indicating that activations from the first, second, and third layers possess equally rich information. Moreover, the stable accuracy scores of the linear probability models indicate that there is a solid linear relationship between the activations and true labels.

Figure 4: Accuracy of Probes on Test Data after PCA



The upper-left graph is the linear probability models, the upper-right is the logistic models, and the bottom-left is the neural networks.

8 Conclusion

In this study, I investigated how the internal representations of neural networks in binary image classification change depending on the depth of layers and training amount. I discovered that the layers later in the network pipeline possess representations with higher multicollinearity. However, the multicollinearity in every layer declines and converges to certain points of value as the amount of training increases. Further regression analysis on the data after PCA revealed that the representations from different layers possess the equivalent level of information about the true binary labels.

My research indicates that probing techniques are a suitable approach to understand *how AI thinks* by quantifying what kind of representations it acquires. The results also present answers to the research questions: (1) every layer contains equally rich information, and (2) the number of training steps does not affect accuracy or heteroskedasticity but makes multicollinearity converge.

This research contributes to the further understanding of deep neural network models not only through probing but also via statistical analysis. Limitations posed to this work are that I solely focused on neural networks for binary image classification. I also failed to replicate similar results with Alain and Bengio [3], who argued that deeper layers contain richer representations in binary image classification. Thus, further studies should be done to apply my analysis methods to neural networks with broader tasks and perform stricter replication studies.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Mohammad Sadegh Akhondzadeh, Vijay Lingam, and Aleksandar Bojchevski. Probing graph representations. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 11630–11649. PMLR, 25–27 Apr 2023.

[3] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2018.

[4] Vy Anh Tran, Le Thi Nhu Quynh, Thu-Thao Thi Vo, Phuc An Nguyen, Ta Ngoc Don, Yasser Vasseghian, Hung Phan, and Sang-Wha Lee. Experimental and computational investigation of a green knoevenagel condensation catalyzed by zeolitic imidazolate framework-8. *Environmental Research*, 204:112364, 2022.

[5] Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 04 2022.

[6] Prashant Gohel, Priyanka Singh, and Manoranjan Mohanty. Explainable ai: current status and future directions, 2021.

[7] Justin Grimmer, Margaret E. Roberts, and Brandon M. Stewart. Machine learning for social science: An agnostic approach. *Annual Review of Political Science*, 24, 2021.

[8] Wes Gurnee and Max Tegmark. Language models represent space and time, 2024.

[9] Gabriel Ilharco, Rowan Zellers, Ali Farhadi, and Hannaneh Hajishirzi. Probing contextual language models for common ground with visual representations. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5367–5377, Online, June 2021. Association for Computational Linguistics.

[10] Yacine Izza, Alexey Ignatiev, and Joao Marques-Silva. On explaining decision trees, 2020.

[11] Alex Krizhevsky. pages 32–33.

[12] Michael B. McCoy and Anna Leshinskaya. Relational composition during attribute retrieval in GPT is not purely linear. In *NeurIPS 2024 Workshop on Compositional Learning: Perspectives, Methods, and Paths Forward*, 2024.

[13] L.E. Melkumova and S.Ya. Shatskikh. Comparing ridge and lasso estimators for data analysis. *Procedia Engineering*, 201:746–755, 2017. 3rd International Conference "In-

formation Technology and Nanotechnology", ITNT-2017, 25-27 April 2017, Samara, Russia.

[14] Jerry Ngo and Yoon Kim. What do language models hear? probing for auditory representations in language models, 2024.

[15] Saeed Nosratabadi, Amirhosein Mosavi, Puhong Duan, Pedram Ghamisi, Ferdinand Filip, Shahab S. Band, Uwe Reuter, Joao Gama, and Amir H. Gandomi. Data science in economics: Comprehensive review of advanced machine learning and deep learning methods. *Mathematics*, 8(10), 2020.

[16] Lexi V Perez. Principal component analysis to address multicollinearity. *Whitman College: Walla Walla, WA, USA*, 99362, 2017.

[17] Cyrus Rashtchian, Charles Herrmann, Chun-Sung Ferng, Ayan Chakrabarti, Dilip Krishnan, Deqing Sun, Da-Cheng Juan, and Andrew Tomkins. Probing the equivariance of image embeddings. In *NeurIPS 2023 Workshop on Distribution Shifts: New Frontiers with Foundation Models*, 2024.