OPT2024: 16th Annual Workshop on Optimization for Machine Learning

Hierarchical Simplicity Bias of Neural Networks

Zhehang Du DUZ@WHARTON.UPENN.EDU
The Wharton School, University of Pennsylvania

Abstract

Neural networks often exhibit simplicity bias, favoring simpler features over more complex ones, even when both are equally predictive. We introduce a novel method called *imbalanced label coupling* to explore and extend this simplicity bias across multiple hierarchical levels. Our approach demonstrates that trained networks sequentially consider features of increasing complexity based on their correlation with labels in the training set, regardless of their actual predictive power. For example, in CIFAR-10, simple spurious features can cause misclassifications where most cats are predicted as dogs and most trucks as automobiles. We empirically show that last-layer retraining with target data distribution [18] is insufficient to fully recover core features when spurious features perfectly correlate with target labels in our synthetic datasets. Our findings deepen the understanding of the implicit biases inherent in neural networks.

1. Introduction

Neural networks (NNs) have demonstrated remarkable capabilities in learning and generalizing from data, even in over-parameterized settings, a phenomenon known as double descent [29]. Despite their adeptness, neural networks can exhibit vulnerability when faced with real-world challenges like distribution shifts [13] and adversarial attacks [25, 41]. One underlying reason for these vulnerabilities is the *simplicity bias* [17, 31, 37, 43], where gradient descent tends to favor learning simple, strongly correlated features over more complex but robust ones. This preference for simplicity might seem reasonable, as this inductive bias naturally reflects the properties of real-world data and leads to generalization [42]. However, it can lead to models that struggle when subjected to distributional shifts and adversarial attacks.

Background. Simplicity bias is the tendency of neural networks to learn simple functions, potentially ignoring more complex but equally or more predictive features [37, 43]. This phenomenon is illustrated in the work by Shah et al. [37], where neural networks were shown to preferentially learn from simpler, more salient features at the expense of more complex but equally predictive ones. For example, in their MNIST-CIFAR dataset, images in class -1 are concatenations of MNIST digit zero and CIFAR-10 automobiles, while images in class 1 are concatenations of MNIST digit one and CIFAR-10 trucks. It turns out that the trained network only depends on the MNIST digit for classification. Another example is found in shortcut learning [8], where the neural network focuses on object location rather than object type.

Motivation. Understanding how neural networks learn in the presence of spurious features is critical, especially when these features introduce false associations with target labels, degrading model performance [34]. Previous studies have focused on mitigating these vulnerabilities, but the hierarchical nature of simplicity bias across multiple feature complexities remains underexplored.

Contributions. In this study, we introduce *imbalanced label coupling* to extend the concept of simplicity bias to multiple hierarchical levels. Our key contributions are: (a) We demonstrate that neural networks exhibit a *hierarchical simplicity bias*, making predictions by sequentially considering features of increasing complexity, akin to decision trees. (b) We provide empirical evidence using synthetic datasets designed to highlight hierarchical decision-making based on feature complexity. (c) We show that last-layer retraining with target data distribution [18] is insufficient to fully recover core features when spurious features perfectly correlate with target labels, highlighting limitations in addressing hierarchical simplicity bias.

2. Formulation of Hierarchical Simplicity Bias

Notation. We formally define the notations used in our formulation: $\mathbf{x} = (\mathbf{x}_s, \mathbf{x}_c)$ represents the input data, where $\mathbf{x}_s$ denotes simple features (e.g., MNIST digits or patches), and $\mathbf{x}_c$ denotes complex features (e.g., CIFAR-10 images). The symbol $y \in \mathcal{Y}$ indicates the true label associated with $\mathbf{x}$, while $\hat{y} \in \mathcal{Y}$ represents the predicted label by the neural network. The function $f : \mathcal{X}_s \times \mathcal{X}_c \to \mathcal{Y}$ is the neural network's prediction function. Furthermore, $\mathcal{Y}_{\mathbf{x}_s} \subseteq \mathcal{Y}$ specifies the subset of labels associated with the simple feature $\mathbf{x}_s$ due to imbalanced label coupling.

2.1. Hierarchical Decision Process

We introduce an idealized hierarchical decision process where neural networks perform predictions akin to a decision tree, sequentially considering features of increasing complexity.

Training: Imbalanced Label Coupling. We construct the training set by coupling classes from two different datasets in an imbalanced manner. For each class from the coarse dataset (e.g., MNIST digits), we concatenate it with multiple classes from the fine dataset (e.g., CIFAR-10 images) to create training examples. The labels are assigned **solely based on the fine dataset**, while the coarse dataset introduces spurious correlations.

Depending on which dataset serves as the fine dataset, we consider two scenarios. In Scenario A, the fine dataset corresponds to the complex features $\mathbf{x}_c$, and the labels are assigned based on $\mathbf{x}_c$. The simple features $\mathbf{x}_s$ (from the coarse dataset) introduce spurious correlations. The true label y is determined by the complex features $\mathbf{x}_c$, i.e., $y = y(\mathbf{x}_c)$. Likewise, in scenario B, the fine dataset is simple features. For example, in scenario A, if MNIST digit 1 is paired with CIFAR-10 classes automobile and cat, then any image containing digit 1 can be labeled as either automobile or cat based on the CIFAR-10 image it is paired with.

Testing: Hierarchical Decision Process. The test set is created by concatenating the image channels from all selected classes in each dataset, **without any coupling constraints**. During testing, the neural network's prediction $\hat{y}$ follows a hierarchical decision process:

- **Scenario A:** The coarse features $\mathbf{x}_s$ are correlated with subsets of labels. For each value of $\mathbf{x}_s$, there is an associated subset $\mathcal{Y}_{\mathbf{x}_s} \subseteq \mathcal{Y}$. The prediction function is:

$$\hat{y} = f(\mathbf{x}_s, \mathbf{x}_c) = f_{\mathbf{x}_s}(\mathbf{x}_c) = \arg\max_{y' \in \mathcal{Y}_{\mathbf{x}_s}} p(y' \mid \mathbf{x}_c), \tag{1}$$

  where $f_{\mathbf{x}_s}$ denotes the decision function conditioned on $\mathbf{x}_s$, mapping $\mathbf{x}_c$ to a label within $\mathcal{Y}_{\mathbf{x}_s}$. First, the network uses the coarse features to narrow down the possible labels to a subset. Then, within this subset, the network uses the fine features to predict the final label $\hat{y}$.

- **Scenario B:** The labels are assigned based on the simple features $\mathbf{x}_s$, which are fully predictive. Due to simplicity bias, the network only depends on the simple features $\mathbf{x}_s$ for prediction. The prediction function then simplifies to:

$$\hat{y} = f(\mathbf{x}_s) = \arg\max_{y' \in \mathcal{Y}} p(y' \mid \mathbf{x}_s). \tag{2}$$

In both scenarios, the decision-making prioritizes simple features, i.e., making decisions according to the ascending complexity of features.

2.2. Quantitative Measures of Hierarchical Simplicity Bias

Given that neural networks trained with gradient descent by empirical risk minimization may not perfectly align with the hierarchical prediction, we introduce quantitative measures to more accurately capture the hierarchical simplicity bias observed in the confusion matrix. We define separate measures for each scenario to account for the differences in how the hierarchical simplicity bias manifests.

- **Scenario A.** For each true label y and associated simple feature $\mathbf{x}_s$, we define the **Hierarchical Classification Accuracy (HCA)** as:

$$\text{HCA}(y) = \frac{\text{Accuracy within } \mathcal{Y}_{\mathbf{x}_s} - \text{Chance Accuracy}}{1 - \text{Chance Accuracy}}, \tag{3}$$

where Accuracy within $\mathcal{Y}_{\mathbf{x}_s}$ is defined as:

$$\text{Accuracy within } \mathcal{Y}_{\mathbf{x}_s} = \frac{\arg\max_{y' \in \mathcal{Y}_{\mathbf{x}_s}} \text{CM}_{y',y}}{N_y}. \tag{4}$$

$\text{CM}_{y',y}$ is the count of predictions being y' for samples with true label y, N_y is the total number of samples with true label y, and Chance Accuracy is the accuracy that would be achieved by random guessing within the subset $\mathcal{Y}_{\mathbf{x}_s}$, given by Chance Accuracy $= |\mathcal{Y}_{\mathbf{x}_s}|^{-1}$, where $|\mathcal{Y}_{\mathbf{x}_s}|$ is the number of labels in the subset $\mathcal{Y}_{\mathbf{x}_s}$. This formulation of Accuracy within $\mathcal{Y}_{\mathbf{x}_s}$ captures the maximum accuracy achieved by the most frequently predicted label within $\mathcal{Y}_{\mathbf{x}_s}$ for samples with true label y. By comparing this accuracy to the chance level, we assess whether the network's performance exceeds what would be expected by random guessing within the subset.

Then, We compute the **Average Hierarchical Classification Accuracy (AHCA)** over all true labels:

$$\text{AHCA} = \frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} \text{HCA}(y). \tag{5}$$

- **Scenario B.** In this scenario, the labels are assigned based on the simple features $\mathbf{x}_s$, which are fully predictive of the true labels. The complex features $\mathbf{x}_c$ introduce spurious correlations but should not influence the network's predictions due to simplicity bias. To confirm that the network relies solely on $\mathbf{x}_s$, we define the **Prediction Consistency Score (PCS)** as:

$$\text{PCS} = \frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} \frac{\text{CM}_{y,y}}{N_y}. \tag{6}$$

A high AHCA indicates that the network consistently uses the complex features $\mathbf{x}_c$ to make accurate predictions within each group defined by $\mathbf{x}_s$. A high PCS indicates that the network's predictions are accurate and consistent with the true labels determined by $\mathbf{x}_s$, suggesting that $\mathbf{x}_c$ does not adversely affect the network's decision-making process. To make the experiment more interpretable, we present both measures (AHCA and PCS) for each scenario in the experiments. This approach ensures that the roles of complex features and simple features can not be interchanged, thereby accurately capturing the hierarchical simplicity bias in different experimental setups.

3. Experiment

3.1. Experiment Setup

Building blocks. We utilize three primary datasets to construct our synthetic datasets: **(1)** Patch: Four types of deterministic patches, each featuring a white corner with the remaining area in black (Figure 1). The patch data is deterministic. **(2)** MNIST [22]. **(3)** CIFAR-10 [20].

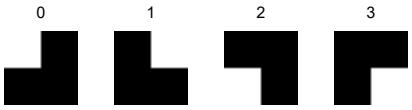

Figure 1: The patch data types.

Synthetic Datasets. We create four training datasets by combining the building blocks using imbalanced label coupling, as shown in Figure 2. Details of the experimental setup, including data preprocessing and model configurations, are provided in Appendix B.

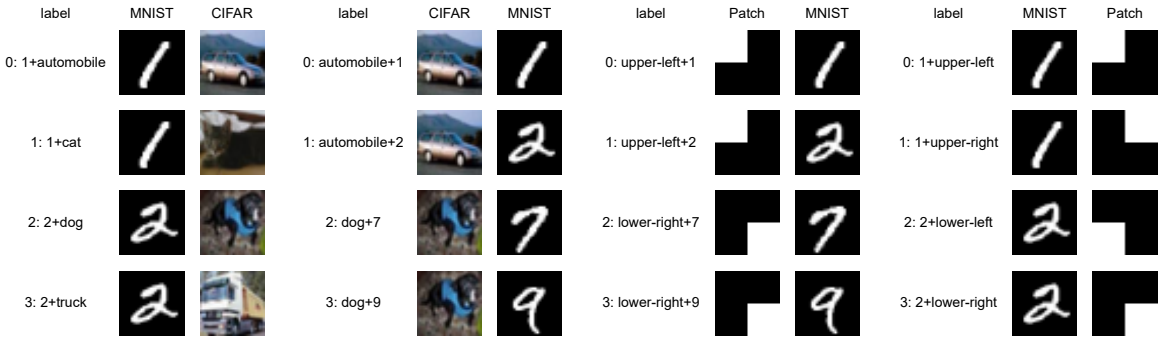

Figure 2: Illustration of four datasets. From left to right: (a) MNIST-CIFAR, (b) CIFAR-MNIST, (c) Patch-MNIST, (d) MNIST-Patch.

3.2. Results and Discussion

In the MNIST-CIFAR dataset shown in Figure 2(a), we combine two datasets: CIFAR-10 images and MNIST digits. Here, CIFAR-10 images ($\mathbf{x}_c$) represent the complex features (fine dataset), while MNIST digits ($\mathbf{x}_s$) represent the simple features (coarse dataset). Each MNIST digit is linked to a specific subset of CIFAR-10 labels; for example, the digit 1 corresponds to $\mathcal{Y}_{\mathbf{x}_s} =$

$\{\text{automobile}, \text{cat}\}$. As shown in Figure 3, the trained neural network exhibits a hierarchical decision-making process akin to a decision tree. Initially, the network uses the simple MNIST digit ($\mathbf{x}_s$) to narrow down the possible labels to a subset. Then, within this subset, it uses the complex CIFAR-10 image ($\mathbf{x}_c$) for fine-grained classification.

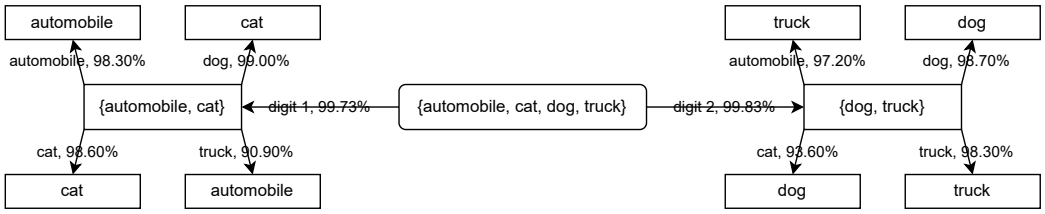

Figure 3: The inferred decision tree from the neural network trained on the MNIST-CIFAR dataset. The boxes contain the network's predictions, while the arrows indicate the path taken based on the input features. The percentages show the proportion of samples following each path. For example, among all samples with MNIST digit 1, approximately 99.73% are predicted to be either automobile or cat. Within this group, 99.00% of samples where the CIFAR-10 image is a dog are misclassified as cat.

Notably, when misclassifications occur, they are not random but instead exhibit patterns that reflect the retained predictive power of the complex features. For instance, many images of automobiles are misclassified as trucks, and cats are misclassified as dogs. These specific errors suggest that the network is leveraging semantic similarities among CIFAR-10 classes within the subsets determined by the MNIST digits.

To quantitatively assess the hierarchical simplicity bias, we calculate the Average Hierarchical Classification Accuracy (AHCA) and Prediction Consistency Score (PCS) for each dataset, as shown in Table 4(a). Recall that AHCA measures the network's ability to use complex features within groups defined by simple features, while PCS measures the reliance on simple features when they are fully predictive.

Dataset	AHCA	PCS
MNIST-CIFAR	93.65	49.35
CIFAR-MNIST	0.93	99.24
Patch-MNIST	78.05	49.96
MNIST-Patch	0.00	100.00

Model	Standard	Semantic
Spurious	49.24	96.83
DFR	68.12	97.55
Baseline	87.75	98.30

Figure 4: (a) Left: AHCA and PCS (%) for four different datasets. (b) Right: Performance comparison of three models: Spurious (trained on the MNIST-CIFAR dataset), DFR (the Spurious model after DFR), and Baseline (trained on CIFAR-10 images).

In scenarios where labels are based on complex features (e.g., *MNIST-CIFAR* and *Patch-MNIST*), the high AHCA values indicate that networks effectively utilize complex features for classification within the groups defined by simple features. Conversely, in scenarios where labels are based on simple features (e.g., *CIFAR-MNIST* and *MNIST-Patch*), the high PCS values and low AHCA val-

ues illustrate that networks predominantly depend on simple features and largely ignore complex features. More detailed experiment results are in Appendix C.

Building on our observations of hierarchical simplicity bias, we investigate whether existing methods can mitigate the impact of spurious correlations in such hierarchical settings.

3.3. Last-Layer Retraining Is Insufficient for Strong Spurious Correlations

Kirichenko et al. [18] introduced Deep Feature Reweighting (DFR), a method aimed at improving model robustness against spurious correlations by retraining the last layer using data from the target distribution. We apply DFR to our MNIST-CIFAR dataset (Figure 2(a)) to assess its effectiveness in our hierarchical setting. In this dataset, classes 0 and 3 correspond to vehicles (automobile and truck), and classes 1 and 2 correspond to animals (dog and cat). We define a *semantically correct prediction* as correctly classifying a vehicle as a vehicle or an animal as an animal, regardless of the specific class label. It allows us to evaluate whether the network retains an understanding of core features at a higher level of abstraction. The results are presented in Table 4(b).

Although DFR improves the standard accuracy from 49.24% to 68.12%, it does not achieve the baseline accuracy of 87.75%, suggesting that DFR alone may not fully address the challenges posed by perfectly correlated spurious features. This limitation indicates that some core feature information may not be adequately captured in the final-layer representations, possibly due to loss of information in intermediate layers when spurious features are very strong.

Interestingly, the high semantic accuracies suggest that the neural network retains an understanding of core features at a higher level of abstraction. For the Spurious model, the semantic accuracy is 96.83%, which is very close to the baseline model's 98.30%. This indicates that despite the influence of spurious features on fine-grained class distinctions, the network is still capable of distinguishing between broader categories such as vehicles and animals. This observation underscores the hierarchical nature of the network's decision-making process, where it can process and retain information at multiple levels of complexity. It also highlights the importance of evaluating model performance across different levels of abstraction to gain deeper insights into neural network behavior beyond overall accuracy metrics.

4. Conclusion and Future Work

Conclusion. We introduced *imbalanced label coupling* to extend simplicity bias to multiple hierarchical levels, demonstrating that neural networks exhibit a hierarchical simplicity bias. Our experiments with synthetic datasets show that networks prioritize features based on ascending complexity correlated with labels, mirroring a decision tree's behavior. Moreover, we found that last-layer retraining is insufficient to recover core features when spurious correlations are perfect, indicating that core feature information may be lost in intermediate layers. Our study provides insights into how neural networks prioritize features of varying complexities, contributing to a deeper understanding of implicit bias and aiding the development of more robust machine learning systems.

Limitations and Future Work. The extreme hierarchical relationships observed in our study may not be easily observable in practice, as our research is based on synthetic datasets designed to highlight certain behaviors in a controlled setting. Future work could extend these findings to real-world situations, establish a theoretical framework for hierarchical simplicity bias, and propose strategies to mitigate adverse effects of this bias.

References

[1] Michael A Alcorn, Qi Li, Zhitao Gong, Chengfei Wang, Long Mai, Wei-Shinn Ku, and Anh Nguyen. Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4845–4854, 2019.

[2] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

[3] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. *Advances in Neural Information Processing Systems*, 32, 2019.

[4] Aharon Ben-Tal, Dick Den Hertog, Anja De Waegenaere, Bertrand Melenberg, and Gijs Rennen. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357, 2013.

[5] Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. Towards understanding the spectral bias of deep learning. *arXiv preprint arXiv:1912.01198*, 2019.

[6] Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. Environment inference for invariant learning. In *International Conference on Machine Learning*, pages 2189–2200. PMLR, 2021.

[7] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[8] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[9] Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. *Advances in neural information processing systems*, 30, 2017.

[10] Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. *Advances in neural information processing systems*, 31, 2018.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[13] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021.

[14] Katherine Hermann and Andrew Lampinen. What shapes feature representations? exploring datasets, architectures, and training. *Advances in Neural Information Processing Systems*, 33: 9995–10006, 2020.

[15] Weihua Hu, Gang Niu, Issei Sato, and Masashi Sugiyama. Does distributionally robust supervised learning give robust classifiers? In *International Conference on Machine Learning*, pages 2029–2037. PMLR, 2018.

[16] Minyoung Huh, Hossein Mobahi, Richard Zhang, Brian Cheung, Pulkit Agrawal, and Phillip Isola. The low-rank simplicity bias in deep networks. *arXiv preprint arXiv:2103.10427*, 2021.

[17] Dimitris Kalimeris, Gal Kaplun, Preetum Nakkiran, Benjamin Edelman, Tristan Yang, Boaz Barak, and Haofeng Zhang. Sgd on neural networks learns functions of increasing complexity. *Advances in neural information processing systems*, 32, 2019.

[18] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022.

[19] Masanori Koyama and Shoichiro Yamaguchi. Out-of-distribution generalization with maximal invariant predictor. 2020.

[20] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[21] Daniel Kunin, Atsushi Yamamura, Chao Ma, and Surya Ganguli. The asymmetric maximum margin bias of quasi-homogeneous neural networks. *arXiv preprint arXiv:2210.03820*, 2022.

[22] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]*, 2, 2010. URL http://yann.lecun.com/exdb/mnist.

[23] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR, 2021.

[24] Kaifeng Lyu, Zhiyuan Li, Runzhe Wang, and Sanjeev Arora. Gradient descent on two-layer nets: Margin maximization and simplicity bias. *Advances in Neural Information Processing Systems*, 34:12978–12991, 2021.

[25] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[26] Mazda Moayeri, Phillip Pope, Yogesh Balaji, and Soheil Feizi. A comprehensive study of image classification model sensitivity to foregrounds, backgrounds, and visual attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19087–19097, 2022.

[27] Depen Morwani, Jatin Batra, Prateek Jain, and Praneeth Netrapalli. Simplicity bias in 1-hidden layer neural networks. *arXiv preprint arXiv:2302.00457*, 2023.

[28] Mor Shpigel Nacson, Jason Lee, Suriya Gunasekar, Pedro Henrique Pamplona Savarese, Nathan Srebro, and Daniel Soudry. Convergence of gradient descent on separable data. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3420–3428. PMLR, 2019.

[29] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

[30] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. *Advances in Neural Information Processing Systems*, 33:20673–20684, 2020.

[31] Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *Advances in Neural Information Processing Systems*, 34:1256–1272, 2021.

[32] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pages 5301–5310. PMLR, 2019.

[33] Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization. *arXiv preprint arXiv:2202.06856*, 2022.

[34] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[35] Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pages 8346–8356. PMLR, 2020.

[36] Luca Scimeca, Seong Joon Oh, Sanghyuk Chun, Michael Poli, and Sangdoo Yun. Which shortcut cues will dnns choose? a study from the parameter-space perspective. *arXiv preprint arXiv:2110.03095*, 2021.

[37] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. *Advances in Neural Information Processing Systems*, 33:9573–9585, 2020.

[38] Sahil Singla and Soheil Feizi. Salient imagenet: How to discover spurious features in deep learning? *arXiv preprint arXiv:2110.04301*, 2021.

[39] Sahil Singla, Besmira Nushi, Shital Shah, Ece Kamar, and Eric Horvitz. Understanding failures of deep networks via robust feature extraction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12853–12862, 2021.

[40] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

[41] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[42] Damien Teney, Ehsan Abbasnejad, Simon Lucey, and Anton Van den Hengel. Evading the simplicity bias: Training a diverse set of models discovers solutions with superior ood generalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16761–16772, 2022.

[43] Guillermo Valle-Perez, Chico Q Camargo, and Ard A Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. *arXiv preprint arXiv:1805.08522*, 2018.

[44] Huaxiu Yao, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. Improving out-of-distribution robustness via selective augmentation. In *International Conference on Machine Learning*, pages 25407–25437. PMLR, 2022.

[45] Haotian Ye, James Zou, and Linjun Zhang. Freeze then train: Towards provable representation learning under spurious correlations and feature noise. In *International Conference on Artificial Intelligence and Statistics*, pages 8968–8990. PMLR, 2023.

[46] Jingzhao Zhang, Aditya Menon, Andreas Veit, Srinadh Bhojanapalli, Sanjiv Kumar, and Suvrit Sra. Coping with label shift via distributionally robust optimisation. *arXiv preprint arXiv:2010.12230*, 2020.

[47] Michael Zhang, Nimit S Sohoni, Hongyang R Zhang, Chelsea Finn, and Christopher Ré. Correct-n-contrast: A contrastive approach for improving robustness to spurious correlations. *arXiv preprint arXiv:2203.01517*, 2022.

Appendix A. Related Work

Simplicity bias. Kalimeris et al. [17] show that stochastic gradient descent learns functions of increasing complexity. Gunasekar et al. [10], Kunin et al. [21], Lyu et al. [24], Nacson et al. [28], Soudry et al. [40] show that gradient descent leads to max-margin classifiers. Arora et al. [3], Gunasekar et al. [9], Huh et al. [16] show that deep networks are inductively biased to find low-rank solutions. Valle-Perez et al. [43] show that the parameter-function map is biased towards simple functions with algorithmic information theory. Pezeshki et al. [31] propose gradient starvation. Morwani et al. [27] rigorously study simplicity bias in one hidden layer neural networks. Cao et al. [5], Rahaman et al. [32] study the spectral bias of deep neural networks.

Taking a step further, our study reveals a more nuanced perspective. Complex features are not entirely ignored but can retain their predictive power. Our findings do not contradict prior research where simple features have complete predictive power. Rather, in our experiments, simple features serve as predictive indicators only for the specific subgroups trained with those features. Thus, our work extends earlier research: where extreme simplicity bias is similar to a single-level decision tree, our approach resembles a multi-level decision tree.

Spurious correlations. Neural networks exhibit bias towards spurious features, which are only associated with the task label but are not causally related [8]. Such biases have been found in real-world image datasets [38, 39]. Such features include texture [7], poses [1], and background [7, 26]. Overparametrization's negative impact on model performance in the context of these spurious correlations is investigated by Sagawa et al. [35]. Further, studies have been conducted to understand the influencing factors of feature representations [14] and to identify likely shortcut cues [36]. Mitigation methods include distributionally robust optimization (DRO) targeting worst-group loss instead of average loss [4, 15, 34, 46], invariant learning [2, 6, 19, 44], and weighting [23, 30, 47].

In closely related works, Kirichenko et al. [18], Rosenfeld et al. [33] demonstrate that despite spurious correlations, neural networks capture core features that can be recovered by retraining the final layer with target distribution data. Our work, on the other hand, investigates the confusion matrix, highlighting the network's inherent reliance on true object semantics in the presence of spurious features, without training a feature representation decoder. We also show that last-layer retraining is insufficient to fully capture the network's ability to utilize core features. Similarly, Ye et al. [45] provide a theoretical analysis of last-layer retraining for two-layer networks, proving that core features are only learned well when their associated non-realizable noise is small.

Appendix B. Experiment Setup

Training set. The training set is created through *imbalanced label coupling*. We choose two different datasets: one with coarse labels and another with fine labels. For each class from the coarse dataset, we concatenate it with multiple classes from the fine dataset to create the training examples. We assign labels to these examples based on the fine dataset. In other words, the fine dataset has full predictive power, while the coarse dataset does not. Across all experiments, every class comprises 5000 training examples, and the total number of classes is the number of classes chosen in the fine dataset. This construction can be naturally extended to three or more datasets.

Our patch dataset includes four types of patch data, each sized at 32×32 and 1 channel. The MNIST dataset's images are subjected to zero padding, extending them from their original dimensions of 28×28 to 32×32.

For instance, we form the MNIST-CIFAR dataset by concatenating the CIFAR-10 image with dimensions $3 \times 32 \times 32$ and the expanded MNIST digit with dimensions $1 \times 32 \times 32$ to generate a new image with dimensions $4 \times 32 \times 32$.

Test set. The test set is created by concatenating the image channels from all selected classes in each dataset, without any coupling constraints. Each potential combination results in 1000 test samples. Hence, the overall test sample count equals 1000 multiplied by the product of the selected class counts in each dataset.

Network architecture and training configuration. We use the ResNet-18 architecture [11], only making slight adjustments to align the channel of the initial convolutional filter with the input and output dimensions to match the class count. We train the model for 150 epochs using the SGD optimizer with a momentum of 0.9, employing a batch size of 128. The initial learning rate is 0.1, and is reduced by a factor of 10 at epochs 50 and 100. We apply a weight decay of 0.0005. The loss function used for training is the cross-entropy loss. For more experiments on the multilayer perceptron (MLP), see Appendix E.

Appendix C. Detailed Experiment Results

This section presents the comprehensive results of the experiments outlined in Section 3. We include the confusion matrices for all experimental setups, which are utilized to calculate the Average Hierarchical Classification Accuracy (AHCA) and Prediction Consistency Score (PCS). These metrics provide a quantitative evaluation of the neural networks' hierarchical simplicity bias, illustrating how predictions align with feature complexity and the extent to which simple and complex features influence classification outcomes.

MNIST-CIFAR hierarchy. We use MNIST as the coarse data and CIFAR-10 as the fine data in the MNIST-CIFAR training set, shown in Figure 5. Conversely, the CIFAR-MNIST training set is shown in Figure 6. The neural network consistently gives priority to MNIST regardless of its predictive power. The CIFAR-10 part only comes into play when MNIST's predictive power is insufficient. Hence, MNIST exhibits a strictly simpler complexity than CIFAR-10.

Patch-MNIST Hierarchy. Similarly, we construct the training set of Patch-MNIST and MNIST-Patch. Figure 7 and Figure 8 show the training set and test results. These results highlight the hierarchy that patch data is inherently less complex than MNIST. Interestingly, predictions are not completely random with spurious patch data in Figure 7. For instance, when comparing digits 1 and 2, digit 2 has the closest visual similarity to digits 7 and 9 (and vice versa for digit 7). As a result, a large portion of digits 7 and 9 are categorized under digit 2 rather than digit 1.

Patch-MNIST-CIFAR hierarchy. We construct a dataset named Patch-MNIST-CIFAR, containing all aforementioned building blocks, revealing a three-level hierarchy. The upper-left patch is coupled with digits 1 and 2, while the lower-right patch is coupled with digits 5 and 9. These digits (1, 2, 5, 9) are then coupled with CIFAR-10 classes (0, 1), (2, 3), (4, 5), and (6, 7) respectively, resulting in an 8-class dataset. The results are shown in Figure 9. The neural network demonstrates a distinct hierarchy in its predictions based on increasing feature complexity: first patch data, then MNIST digits, and finally CIFAR-10 images.

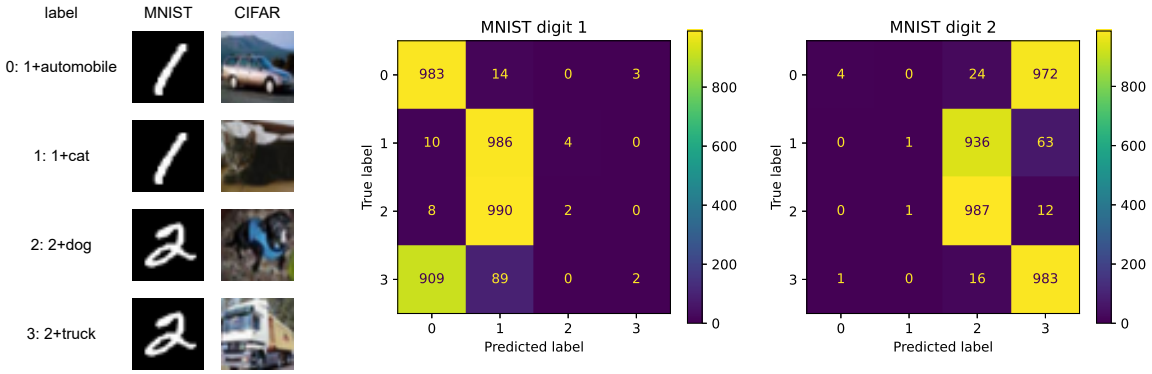

Figure 5: The MNIST-CIFAR training set and test results. **(a)** Left: within the training set, digit 1 is concatenated with both automobile and cat images, while digit 2 is concatenated with both dog and truck images. **(b)** Right: during testing, most digit 1 samples are classified 0 and 1, while most digit 2 samples are classified 2 and 3, regardless of the corresponding CIFAR-10 image class.

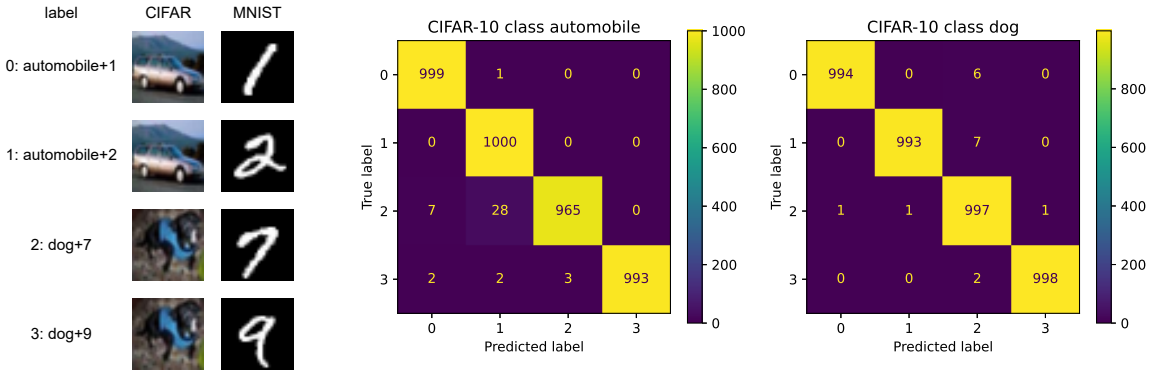

Figure 6: The CIFAR-MNIST training set and test results. **(a)** Left: the training set includes automobile images paired with 1 and 2 digits, along with cat images paired with 7 and 9 digits. **(b)** Right: during testing, predictions rely mostly on MNIST and exhibit high accuracy.

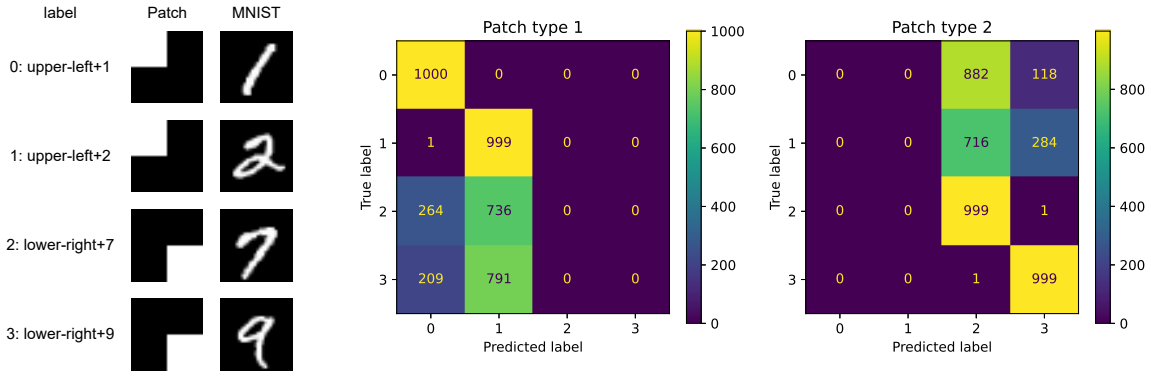

Figure 7: The Patch-MNIST training set and test results. **(a)** Left: the upper-left patch is paired with 1 and 2 digits, along with the lower-right patch paired with 7 and 9 digits. **(b)** Right: in testing, all predictions prioritize patch data without any exceptions.

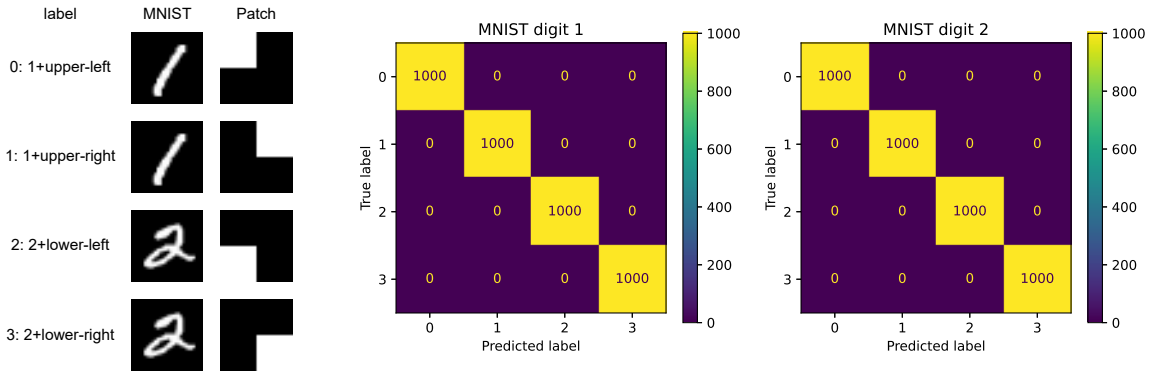

Figure 8: The MNIST-Patch training set and test results. **(a)** Left: digit 1 is paired with upper-left and upper-right patches, and digit 2 is paired with lower-left and lower-right patches. **(b)** Right: the test accuracy in achieves 100% using solely patch data.

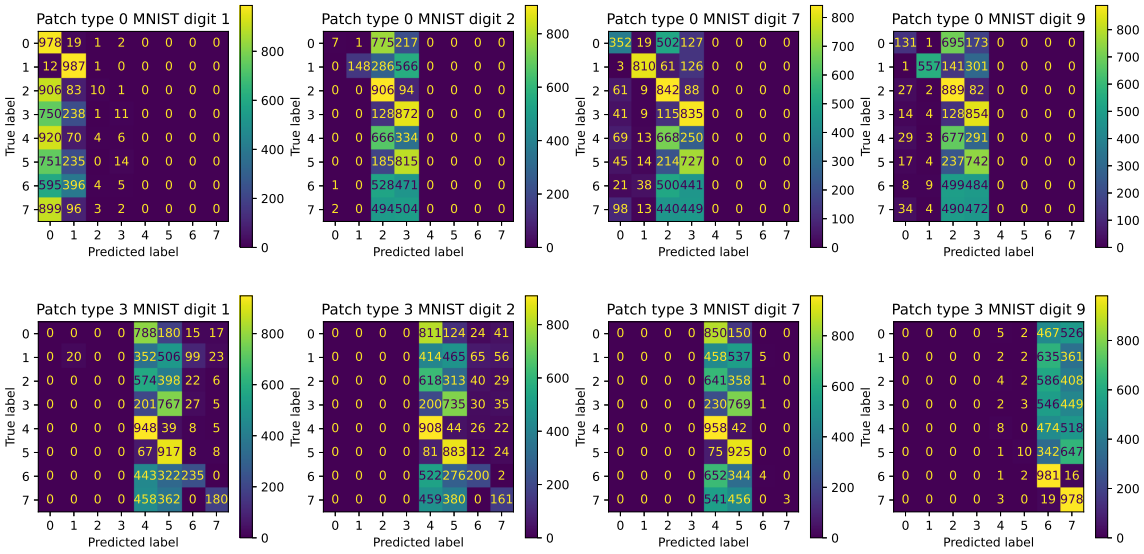

Figure 9: The confusion matrices of the network trained on the Patch-MNIST-CIFAR dataset.

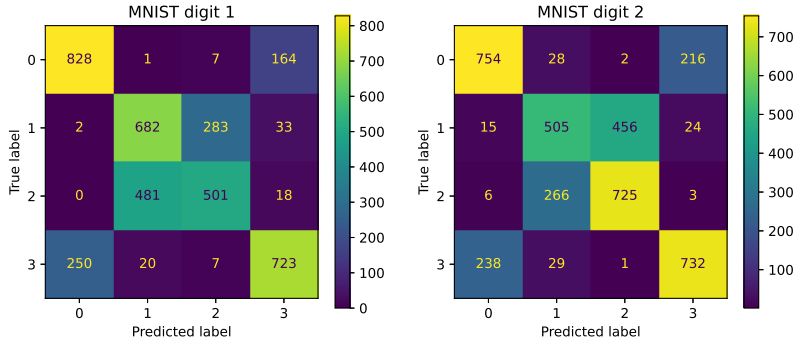

Figure 10: Confusion matrices following DFR of the neural network trained on the MNIST-CIFAR dataset in Figure 2(a).

Appendix D. Background and Corruption Hierarchy

It has been demonstrated that neural networks exhibit a bias for texture and background [7, 26]. In this section, we present similar evidence of decision-tree-like behavior in scenarios involving different background colors and image corruptions.

D.1. Background Hierarchy in Half-Inverted MNIST

During training, we inverted the colors of MNIST digits 0-4 while leaving digits 5-9 unchanged. We tested the model using both the original and color-inverted test sets, with the results shown in Figure 13. The neural network consistently categorized white-background samples as 0-4 and black-background samples as 5-9, even though MNIST digits are nearly linearly separable. Unlike the ColorMNIST dataset in Zhang et al. [47] that uses five different background colors, our study uses just two (and also inverts the color of the digits themselves), which helps to better visualize the hierarchical decision-making process: most occurrences of digit 4 are wrongly classified as 9, and most of digit 3 as 5, and vice versa.

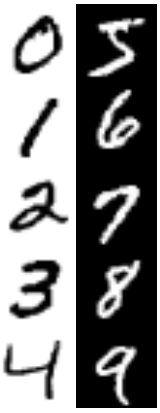

Figure 11: The training set.

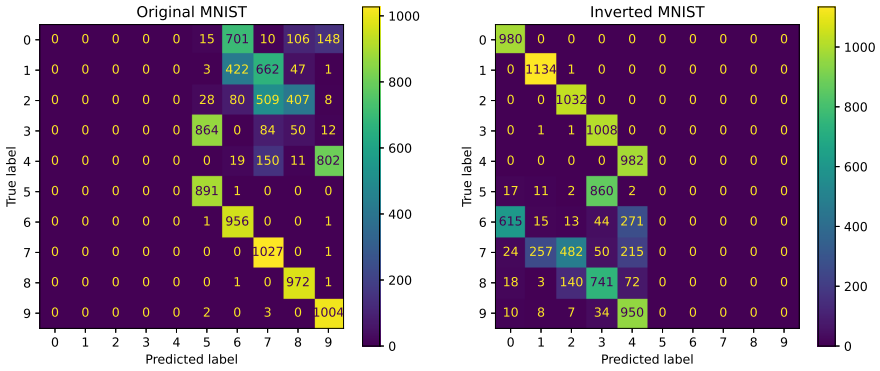

Figure 12: The confusion matrices on the original and color-inverted test set.

Figure 13: The half-inverted MNIST dataset and test results. **(a)** Color inversion applied to digits 0-4; digits 5-9 remain unchanged. **(b)** During testing, the neural network demonstrates a consistent preference for spurious background colors, with digits of similar shapes exhibiting mutual misclassification.

D.2. Corruption Hierarchy in Corrupted CIFAR-10

We subject the CIFAR-10 training set to four common types of corruptions [12]: Gaussian noise, Defocus blur, Fog, and Brightness, all with a severity level of 3. For detailed implementations, see Hendrycks and Dietterich [12]. Each type of corruption is applied to specific CIFAR-10 classes: (0, 1), (2, 3), (4, 5), and (6, 7), while classes (8, 9) remain uncorrupted. The test set includes the original test set with $10,000$ samples, as well as the four corrupted versions of the test set, yielding

a combined dataset of $50,000$ samples. The results are in Figure 14. The hierarchical behavior is not easily visualizable under this setting as there are a total of five groups subjected to different corruptions. Nevertheless, the classification results are not uniformly randomly distributed within each group, which may provide evidence of the hierarchical classification process.

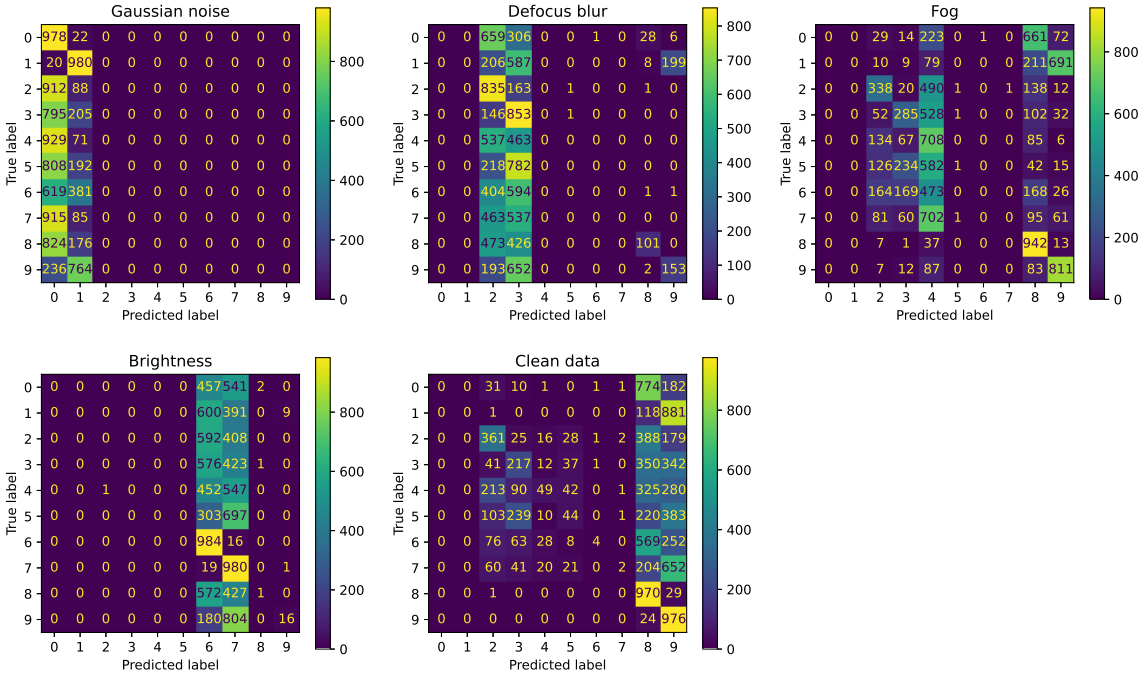

Figure 14: Test results from the neural network trained on the corrupted CIFAR-10 dataset demonstrate that most corruptions heavily influence classification, often leading to predictions falling within the two classes on which they were trained.

Appendix E. Additional Experiments on MLP

To ensure consistency of results across different architectures, we repeated all experiments, except for those in Subsection 3.3, using a multi-layer perceptron (MLP). The MLP has 10 hidden layers, each containing a linear layer with a width of 1024, followed by batch normalization and ReLU activation. The final layer is linear. Other training setups are the same as those in Appendix B. From the results below, we observe the hierarchical decision-making process in MLP, although it is not as pronounced as in the ResNet architecture for the corrupted CIFAR-10 dataset.

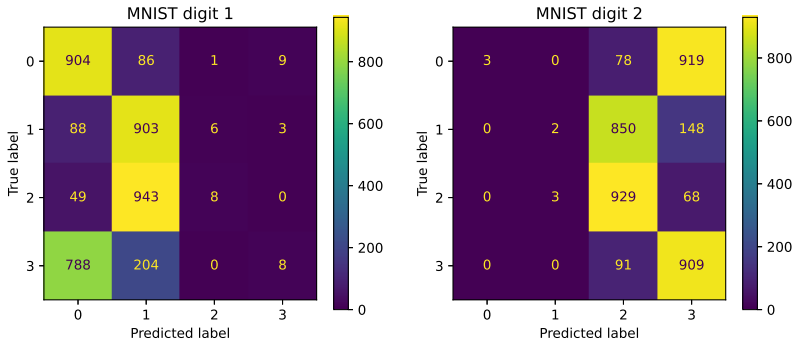

Figure 15: The confusion matrices for the MLP trained on the MNIST-CIFAR dataset.

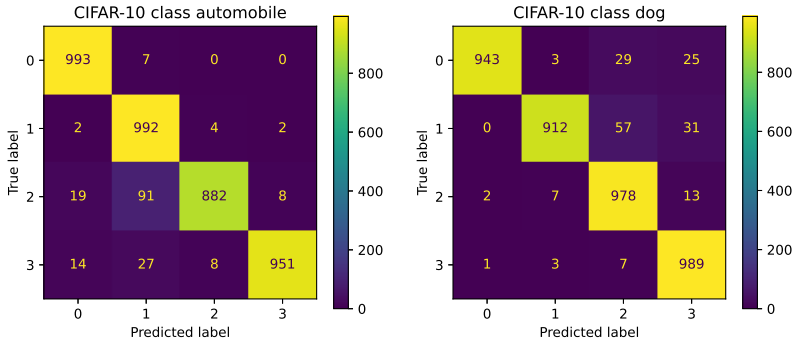

Figure 16: The confusion matrices for the MLP trained on the CIFAR-MNIST dataset.

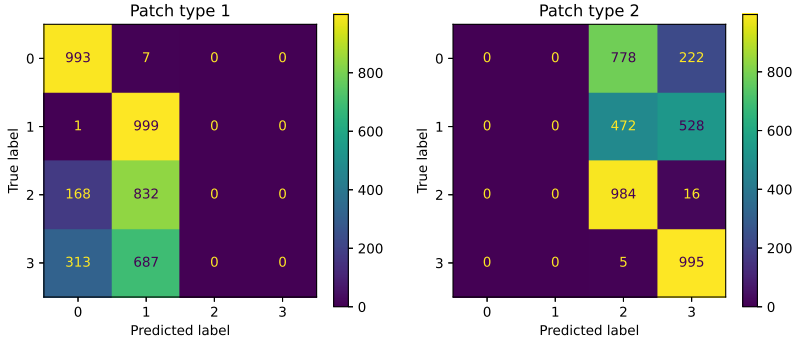

Figure 17: The confusion matrices for the MLP trained on the Patch-MNIST dataset.

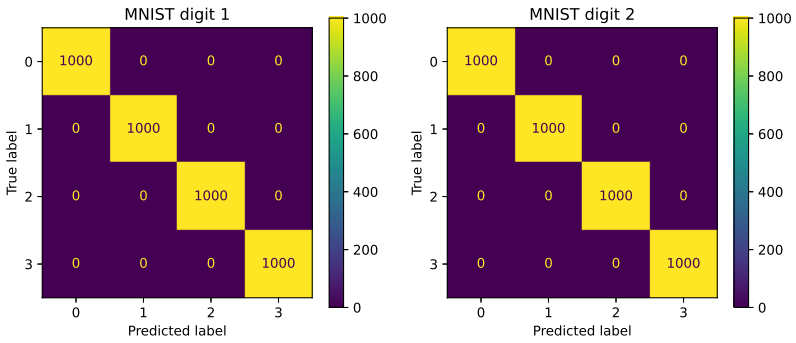

Figure 18: The confusion matrices for the MLP trained on the MNIST-Patch dataset.

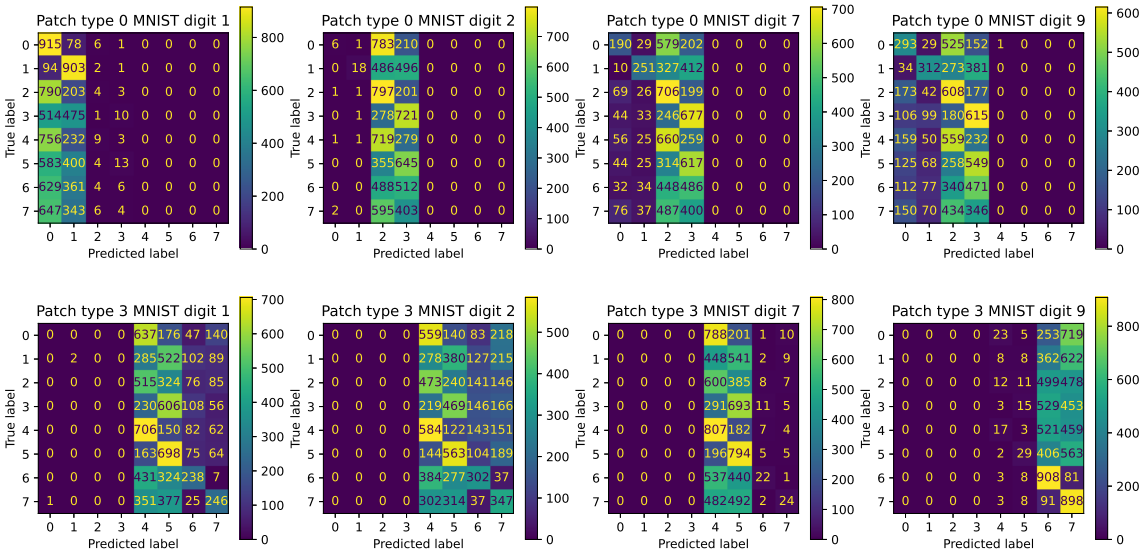

Figure 19: The confusion matrices for the MLP trained on the Patch-MNIST-CIFAR dataset.

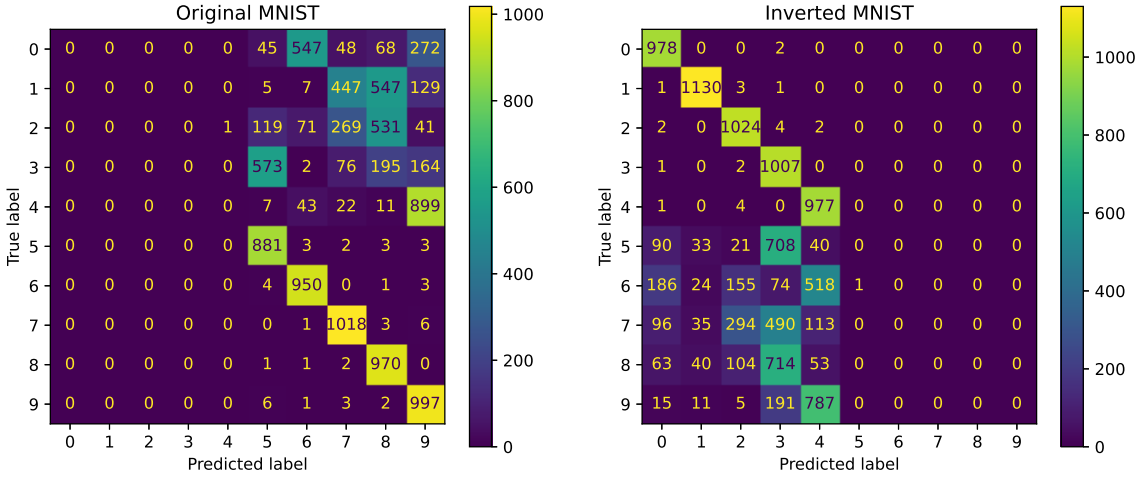

Figure 20: The confusion matrices for the MLP trained on the half-inverted MNIST dataset.

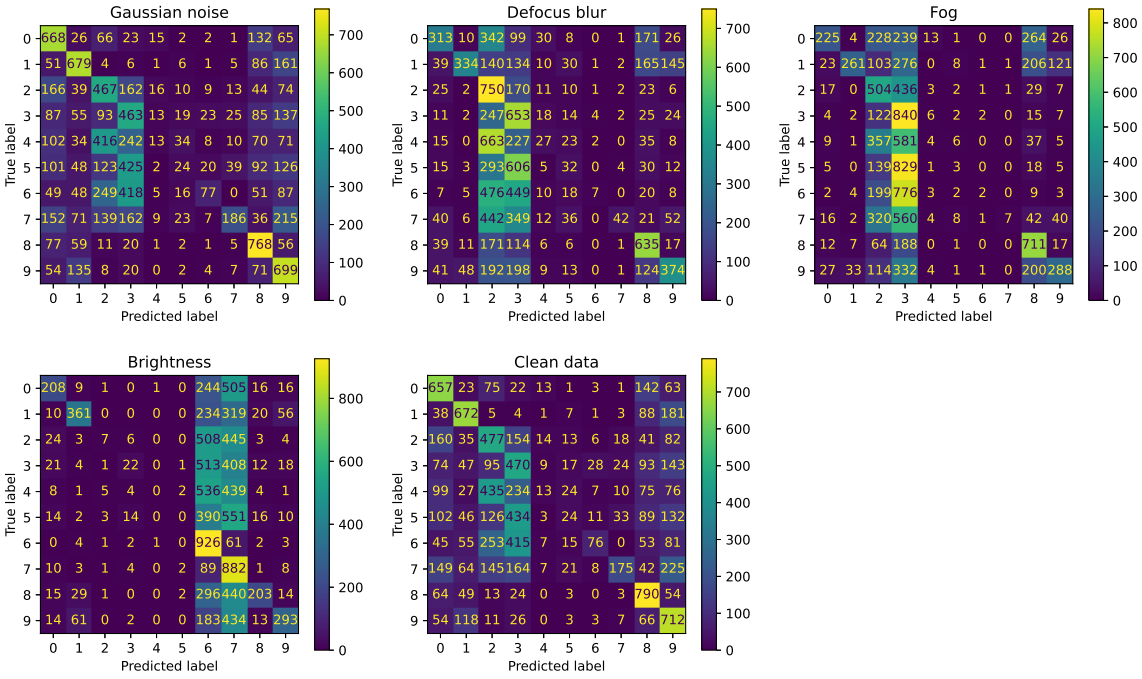

Figure 21: The confusion matrices for the MLP trained on the corrupted CIFAR-10 dataset.