How does a Neural Network's Architecture Impact its Robustness to Noisy Labels?

Anonymous Author(s)
Affiliation
Address
email

Abstract

Noisy labels are inevitable in large real-world datasets. In this work, we explore an area understudied by previous works — how the network's architecture impacts its robustness to noisy labels. We provide a formal framework connecting the robustness of a network to the alignments between its architecture and target/noise functions. Our framework measures a network's robustness via the predictive power in its representations — the test performance of a linear model trained on the learned representations using a small set of clean labels. We hypothesize that a network is more robust to noisy labels if its architecture is more aligned with the target function than the noise. To support our hypothesis, we provide both theoretical and empirical evidence across various neural network architectures and different domains. We also find that when the network is well-aligned with the target function, its predictive power in representations could improve upon state-of-the-art (SOTA) noisy-label-training methods in terms of test accuracy and even outperform sophisticated methods that use clean labels.

1 Introduction

Supervised learning starts with collecting labeled data. Yet, high-quality labels are often expensive. To reduce annotation cost, we collect labels from non-experts [1–4] or online queries [5–7], which are inevitably noisy. To learn from these noisy labels, previous works propose many techniques, including designing robust losses [8–11], adjusting loss before gradient updates [12–20], selecting trust-worthy samples [9, 19, 21–28], applying robust regularization in training [29–34], meta-learning to avoid over-fitting [35, 36], and semi-supervised learning [25, 37–40] to learn better representations.

While these methods improve some networks' robustness to noisy labels, we observe that their effectiveness depends on how well the network's architecture aligns with the target/noise functions, and they are less effective when encountering more realistic label noise that is class-dependent or instance-dependent. This motivates us to investigate an understudied topic: how the network's architecture impacts its robustness to noisy labels.

We formally answer this question by analyzing how a network's architecture aligns with the target function and the noise. To start, we measure the robustness of a network via the predictive power in its learned representations (Definition 1), as models with large test errors may still learn useful predictive hidden representations [41, 42]. Intuitively, the predictive power measures how well the representations can predict the target function. In practice, we measure it by training a linear model on top of the learned representations using a small set of clean labels and evaluate the linear model's test performance [43].

We find that a network having a more aligned architecture with the target function is more robust to noisy labels due to its more predictive representations, whereas a network having an architecture more aligned with the noise function is less robust. Intuitively, a *good* alignment between a network's

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

architecture and a function exists if the architecture can be decomposed into several modules such that each module can simulate one part of the function with a *small* sample complexity. The formal definition of alignment is in Section 2.3, adapted from [44].

Our proposed framework provides initial theoretical support for our findings on a simplified noisy setting (Theorem 2). Empirically, we validate our findings on synthetic graph algorithmic tasks by designing several variants of Graph Neural Networks (GNNs), whose theoretical properties and alignment with algorithmic functions have been well-studied [44–46]. Many noisy label training methods are applied to image classification datasets, so we also validate our findings on image domains using different architectures.

Most of our analysis and experiments use standard neural network training. Interestingly, we find similar results when using DivideMix [38], a SOTA method for learning with noisy labels: for networks less aligned with the target function, the SOTA method barely helps and sometimes even hurts test accuracy; whereas for more aligned networks, it helps greatly.

For well-aligned networks, the predictive power of their learned representation could further improve the test performance of SOTA methods, especially on class-dependent or instance-dependent label noise where current methods on noisy label training are less effective. Moreover, on Clothing1M [47], a large-scale dataset with real-world label noise, the predictive power of a well-aligned network's learned representations could even outperform some sophisticated methods that use clean labels.

1.1 Related Work

A commonly studied type of noisy label is the random label noise, where the noisy labels are drawn i.i.d. from a uniform distribution. While neural networks trained with random labels easily overfit [48], it has been observed that networks learn simple patterns first [41], converge faster on downstream tasks [42], and benefit from memorizing atypical training samples [49].

Accordingly, many recent works on noisy label training are based on the assumption that when trained with noisy labels, neural networks would first fit to clean labels [9, 22, 23, 38, 39] and learn useful feature patterns [15, 50–52]. Yet, these methods are often more effective on random label noise than on more realistic label noise (i.e., class-dependent and instance-dependent label noise).

Many works on representation learning have investigated the features preferred by a network during training [41, 53–55], and how to interpret the learned representations on clean data [43, 53, 56, 57]. Our paper focuses more on the predictive power rather than the explanatory power in the learned representations. We adapt the method in [43] to measure the predictive power in representations, and we study learning from noisy labels rather than from a clean distribution.

On noiseless settings, prior works show that neural networks have the inductive bias to learn simple patterns [41, 53–55]. Our work formalizes what is considered as simple pattern for a given network via architectural alignments, and we extend the definition of alignment in [44] to noisy settings.

2 Overview of Our Framework

In this section, we introduce our problem settings, give formal definitions for "predictive power" and "alignment," and present our main hypothesis as well as our main theorem.

2.1 Problem Settings

Let $\mathcal{X}$ denote the input domain, which can be vectors, images, or graphs. The task is to learn an underlying target function $f : \mathcal{X} \to \mathcal{Y}$ on a noisy training dataset $S := \{(\boldsymbol{x}_i, y_i)\}_{i\in\mathcal{I}} \bigcup \{(\boldsymbol{x}_i, \hat{y}_i)\}_{i\in\mathcal{I}'}$, where $y := f(\boldsymbol{x})$ denotes the true label for an input $\boldsymbol{x}$, and $\hat{y}$ denotes the noisy label. Here, the set $\mathcal{I}$ contains indices with clean labels, and $\mathcal{I}'$ contains indices with noisy labels. We denote $\frac{|\mathcal{I}'|}{|S|}$ as the *noise ratio* in the dataset S. We consider both regression and classification problems.

Regression settings. We consider a label space $\mathcal{Y} \subseteq \mathbb{R}$ and two types of label noise: a) **additive label noise** [58]: $\hat{y} := y + \epsilon$, where ϵ is a random variable independent from $\boldsymbol{x}$; b) **instance-dependent label noise**: $\hat{y} := g(\boldsymbol{x})$ where $g : \mathcal{X} \to \mathcal{Y}$ is a noise function dependent on the input.

Classification settings. We consider a discrete label space with C classes: $\mathcal{Y} = \{1, 2, \cdots, C\}$, and three types of label noise: a) **uniform label noise**: $\hat{y} \sim \text{Unif}(1, C)$, where the noisy label is drawn from a discrete uniform distribution with values between 1 and C, and thus is independent of the true label; b) **flipped label noise**: $\hat{y}$ is generated based on the value of the true label y and does not

consider other input structures; c) **instance-dependent label noise**: $\hat{y} := g(\boldsymbol{x})$ where $g : \mathcal{X} \to \mathcal{Y}$ is a function dependent on the input $\boldsymbol{x}$'s internal structures. Uniform and flipped label noise are commonly studied by previous works on noisy label learning. A few recent works [59, 60] explore the instance-dependent label noise as it is more realistic.

2.2 Predictive Power in Representations

A network' robustness is often measured by its test performance after training with noisy labels. Yet, since models with large test errors may still learn useful representations, we measure the robustness of a network by how good the learned representations are at predicting the target function — the predictive power in representations. To formalize this definition, we decompose a neural network $\mathcal{N}$ into different modules $\mathcal{N}_1, \mathcal{N}_2, \cdots$, where each module can be a single layer (e.g., a convolutional layer) or a block of layers (e.g., a residual block).

Definition 1. *(Predictive power).* Let $f : \mathcal{X} \to \mathcal{Y}$ denote the underlying target function where the input $\boldsymbol{x} \in \mathcal{X}$ is drawn from a distribution $\mathcal{D}$. Let $\mathcal{C} := \{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$ denote a small set of clean data (i.e., $y_i = f(\boldsymbol{x}_i)$). Given a network $\mathcal{N}$ with n modules $\mathcal{N}_j$, let $h^{(j)}(\boldsymbol{x})$ denote the representation from module $\mathcal{N}_j$ on the input $\boldsymbol{x}$ (i.e., the output of $\mathcal{N}_j$). Let L denote the linear model trained with the clean set $\mathcal{C}$ where we use $h^{(j)}(\boldsymbol{x})$ as the input, and y_i as the target value during training. Then the predictive power of representations from the module $\mathcal{N}_i$ is defined as

$$P_j(f, \mathcal{N}, \mathcal{C}) = \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{D}} \left[l \left(f(\boldsymbol{x}), L(h^{(j)}(\boldsymbol{x})) \right) \right], \tag{1}$$

where l is a loss function used to evaluate the test performance on the learning task.

Notice that smaller $P_j(f, \mathcal{N}, \mathcal{C})$ indicates better predictive power; i.e., the representations are better at predicting the target function. We empirically evaluate the predictive power using linear regression to obtain a trained linear model L, which avoids the issue of local minima as we are solving a convex problem; then we evaluate L on the test set.

2.3 Formalization of Alignment

Our analysis stems from the intuition that a network would be more robust to noisy labels if the target function is easier to learn than the noise function. Thus, we use architectural alignment to formalize what is easy to learn by a given network. Xu et al. [44] define the alignment between a network and a deterministic function via a sample complexity measure (i.e., the number of samples needed to ensure low test error with high probability) in a PAC learning framework. Intuitively, a network aligns well with a function if each module of the network can easily learn one part of the function with a small sample complexity.

Definition 2. *(Alignment, simplified based on Xu et al. [44]).* Let $\mathcal{N}$ denote a neural network with n modules $\mathcal{N}_j$. Given a function $f : \mathcal{X} \to \mathcal{Y}$ which can be decomposed into n functions f_j (e.g., $f(\boldsymbol{x}) = f_1(f_2(...f_n(\boldsymbol{x})))$), the alignment between the network $\mathcal{N}$ and f is defined via

$$\textit{Alignment}(\mathcal{N}, f, \epsilon, \delta) := \max_j \mathcal{M}_{A_j}(f_j, \mathcal{N}_j, \epsilon, \delta), \tag{2}$$

where $\mathcal{M}_{A_j}(f_j, \mathcal{N}_j, \epsilon, \delta)$ denotes the sample complexity measure for $\mathcal{N}_j$ to learn f_j with ϵ precision at a failure probability δ under a learning algorithm A_j.

Remark. Notice that smaller *Alignment*$(\mathcal{N}, f, \epsilon, \delta)$ indicates better alignment between network $\mathcal{N}$ and function f. If f is obtuse or does not have a structural decomposition, we can choose $n = 1$, and the definition of alignment degenerates into the sample complexity measure for $\mathcal{N}$ to learn f. Also, as it is often non-trivial to compute the exact alignment for functions without clear algorithmic structures [44], we use *alignment* more as a qualitative rather than quantitative measure in this paper.

Xu et al. [44] further prove that better alignment implies better sample complexity and vice versa.

Theorem 1. *(Informal; [44]) Fix ϵ and δ. Given a target function $f : \mathcal{X} \to \mathcal{Y}$ and a network $\mathcal{N}$, suppose $\{x_i\}_{i=1}^M$ are i.i.d. samples drawn from a distribution $\mathcal{D}$, and let $y_i := f(x_i)$. Then Alignment$(\mathcal{N}, f, \epsilon, \delta) \leq M$ if and only if there exists a learning algorithm A such that*

$$\mathbb{P}_{x \sim \mathcal{D}} \left[\| f_{\mathcal{N},A}(x) - f(x) \| \leq \epsilon \right] \geq 1 - \delta, \tag{3}$$

where $f_{\mathcal{N},A}$ is the function generated by A on the training data $\{x_i, y_i\}_{i=1}^M$.

Remark. Intuitively, a function f (with a decomposition $\{f_j\}_j$) can be efficiently learned by a network $\mathcal{N}$ (with modules $\{\mathcal{N}_j\}_j$) iff each f_j can be efficiently learned by $\mathcal{N}_j$.

We further extend Definition 2 to work with a random process $\mathcal{F}$ (i.e., a set of all possible sample functions that describes the noisy label distribution).

Definition 3. *(Alignment, extension to various noise functions).* Given a neural network $\mathcal{N}$ and a random process $\mathcal{F}$, for each $f \in \mathcal{F}$, the alignment between $\mathcal{N}$ and f is measured via $\max_j \mathcal{M}_{A_j}(f_j, \mathcal{N}_j, \epsilon, \delta)$ based on Definition 2. Then the alignment between $\mathcal{N}$ and $\mathcal{F}$ is defined as

$$\textit{Alignment}^*(\mathcal{N}, \mathcal{F}, \epsilon, \delta) := \sup_{f \in \mathcal{F}} \max_j \mathcal{M}_{A_j}(f_j, \mathcal{N}_j, \epsilon, \delta),$$

where $\mathcal{N}$ can be decomposed differently for various f.

2.4 Better Alignment Implies Better Robustness (Better Predictive Power)

Building on the definitions of *predictive power* and *alignment*, we hypothesize that a network better-aligned with the target function (smaller *Alignment*$(\mathcal{N}, f, \epsilon, \delta)$) would learn more predictive representations (smaller $P_i(f, \mathcal{N}, \mathcal{C})$) when trained on a given noisy dataset.

Hypothesis 1. (Main Hypothesis). Let $f : \mathcal{X} \to \mathcal{Y}$ denote the target function. Fix ϵ, δ, a learning algorithm A, a noise ratio, and a noise function $g : \mathcal{X} \to \mathcal{Y}$ (which may be a drawn from a random process). Let S denote a noisy training dataset and $\mathcal{C}$ denote a small set of clean data. Then for a network $\mathcal{N}$ trained on S with the learning algorithm A,

$$\textit{Alignment}(\mathcal{N}, f, \epsilon, \delta) \downarrow \Longrightarrow P_j(f, \mathcal{N}, \mathcal{C}) \downarrow, \tag{4}$$

where j is selected based on the network's architectural alignment with the target function (for simplicity, we consider $j = n - 1$ in this work).

We prove this hypothesis for a simplified case where the target function shares some common structures with the noise function (e.g., class-dependent label noise). We refer the readers to Appendix C for a full statement of our main theorem with detailed assumptions.

Theorem 2. *(Main Theorem; informal) For a target function $f : \mathcal{X} \to \mathcal{Y}$ and a noise function $g : \mathcal{X} \to \mathcal{Y}$, consider a neural network $\mathcal{N}$ well-aligned with f such that $P_j(f, \mathcal{N}, \mathcal{C})$ is small when training $\mathcal{N}$ on clean data (i.e., $P_j(f, \mathcal{N}, \mathcal{C}) < c$ for some small constant c). If there exists a function h on the input domain $\mathcal{X}$ such that f and g can be decomposed as follows: $\forall x \in \mathcal{X}$, $f(\boldsymbol{x}) = f_r(h(\boldsymbol{x}))$ with f_r being a linear function, and $g(\boldsymbol{x}) = g_r(h(\boldsymbol{x}))$ for some function g_r, then the representations learned by $\mathcal{N}$ on the noisy dataset still have a good predictive power with $P_j(f, \mathcal{N}, \mathcal{C}) < c$.*

We further provide empirical support for our hypothesis via systematic experiments on various architectures, target and noise functions across both regression and classification settings.

3 Experiments on Graph Neural Networks

We first validate our hypothesis on synthetic graph algorithmic tasks by designing GNNs with different levels of alignments to the underlying target/noise functions. We consider regression tasks. The theoretical properties of GNNs and their alignment with algorithmic regression tasks are well-studied [44–46, 61]. To begin with, we conduct experiments on different types of additive label noise and then extend our experiments to instance-dependent label noise, which is closer to real-life noisy labels.

Common Experimental Settings. The training and validation sets always have the same noise ratio, the percentage of data with noisy labels. We choose mean squared error (MSE) and Mean Absolute Error (MAE) as our loss functions. Due to space limit, the results using MAE are in Appendix A.2. All training details are in Appendix B.2. The test error is measured by mean absolute percentage error (MAPE), a relative error metric.

3.1 Background: Graph Neural Networks

GNNs are structured networks operating on graphs with MLP modules [62, 63]. The input is a graph $\mathcal{G} = (V, E)$ where each node $u \in V$ has a feature vector $\boldsymbol{x}_u$, and we use $\mathcal{N}(u)$ to denote the set of neighbors of u. GNNs iteratively compute the node representations via message passing: (1) the

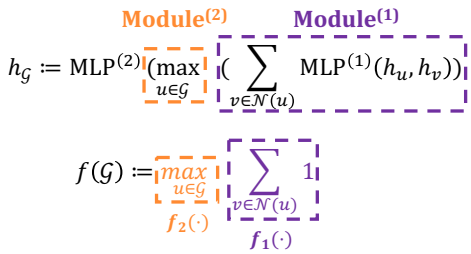

Figure 1: **Max-sum GNN aligns well with the task maximum degree.** Max-sum GNN can be decomposed into two modules: Module$^{(1)}$ and Module$^{(2)}$, and the target function can be similarly divided as $f(\mathcal{G}) = f_2(f_1(\mathcal{G}))$. Note that $f_1(\cdot)$ can be easily learned by Module$^{(1)}$, and $f_2(\cdot)$ is the same as Module$^{(2)}$.

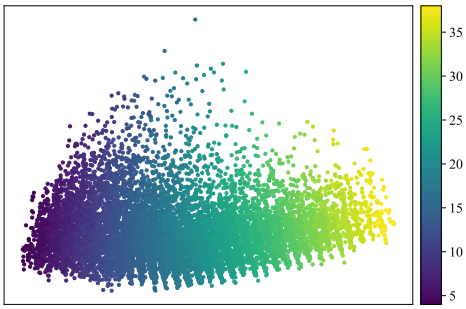

Figure 2: **PCA visualization of hidden representations (colored with true labels)** from a max-sum GNN trained with additive label noise drawn from $\mathcal{N}(10,\ 15)$ at 100% noise ratio. The representations have a clear linear relationship with true labels.

node representation $\boldsymbol{h}_u$ is initialized as the node feature: $\boldsymbol{h}_u^{(0)} = \boldsymbol{x}_u$; (2) in iteration $k = 1..K$, the node representations $\boldsymbol{h}_u^{(k)}$ are updated by aggregating the neighboring nodes' representations with MLP modules [64]. We can optionally compute a graph representation $\boldsymbol{h}_{\mathcal{G}}$ by aggregating the final node representations with another MLP module. Formally,

$$\boldsymbol{h}_u^{(k)} := \sum_{v\in\mathcal{N}(u)} \mathrm{MLP}^{(k)}\left(\boldsymbol{h}_u^{(k-1)}, \boldsymbol{h}_v^{(k-1)}\right), \tag{5}$$

$$\boldsymbol{h}_{\mathcal{G}} := \mathrm{MLP}^{(K+1)}\left(\sum_{u\in\mathcal{G}} \boldsymbol{h}_u^{(K)}\right). \tag{6}$$

Depending on the task, the output is either the graph representation $\boldsymbol{h}_{\mathcal{G}}$ or the final node representations $\boldsymbol{h}_u^{(K)}$. We refer to the neighbor aggregation step for $\boldsymbol{h}_u^{(k)}$ as *aggregation* and the pooling step for $\boldsymbol{h}_{\mathcal{G}}$ as *readout*. Different tasks require different aggregation and readout functions.

3.2 Additive Label Noise

Hu et al. [58] prove that MLPs are robust to additive label noises with zero mean, if the labels are drawn i.i.d. from a Sub-Gaussian distribution. Wu and Xu [65] also show that linear models are robust to zero-mean additive label noise even in the absence of explicit regularization. In this section, we show that a GNN *well-aligned* to the target function not only achieves low test errors on additive label noise with zero-mean, but also learns *predictive* representations on noisy labels that are drawn from non-zero-mean distributions despite having large test error.

Task and Architecture. The task is to compute the maximum node degree:

$$f(\mathcal{G}) := \max_{u\in\mathcal{G}} \sum_{v\in\mathcal{N}(u)} 1. \tag{7}$$

We choose this task as we know which GNN architecture aligns well with this target function—a 2-layer GNN with max-aggregation and sum-readout (max-sum GNN):

$$\boldsymbol{h}_{\mathcal{G}} := \mathrm{MLP}^{(2)}\left(\max_{u\in\mathcal{G}} \sum_{v\in\mathcal{N}(u)} \mathrm{MLP}^{(1)}\left(\boldsymbol{h}_u, \boldsymbol{h}_v\right)\right), \tag{8}$$

$$\boldsymbol{h}_u := \sum_{v\in\mathcal{N}(u)} \mathrm{MLP}^{(0)}\left(\boldsymbol{x}_u, \boldsymbol{x}_v\right). \tag{9}$$

Figure 1 demonstrates how exactly the max-sum GNN aligns with $f(\mathcal{G})$. Intuitively, they are well-aligned as the MLP modules of max-sum GNN only need to learn simple constant functions to simulate $f(\mathcal{G})$. Based on Figure 1, we take the output of Module$^{(2)}$ as the learned representations for max-sum GNNs when evaluating the predictive power.

Label Noise. We corrupt labels by adding independent noise ϵ drawn from three distributions: Gaussian distributions with zero mean $\mathcal{N}(0,\ 40)$ and non-zero mean $\mathcal{N}(10,\ 15)$, and a long-tailed

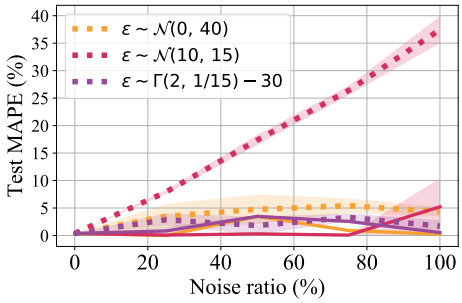

Figure 3: **Representations are very predictive for a GNN well-aligned with the target function under additive label noise.** On the maximum degree task, the representations' predictive power (solid lines) achieves low test MAPE ($< 5\%$) across all three types of noise for the max-sum GNN, despite that the model's test MAPE (dotted lines) may be quite large (for non-zero-mean noise). We average the statistics over 3 runs using different random seeds.

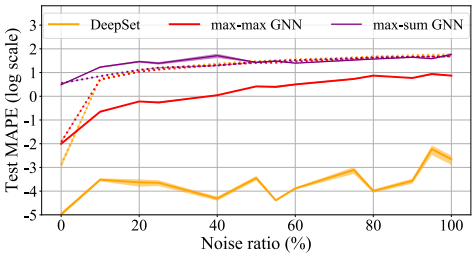

Figure 4: **Representations are more predictive for GNNs more aligned with the target function, and less predictive for GNNs more aligned with the noise function**. On the maximum node feature task, while all three GNNs have large test errors under high noise ratios (dotted lines), the predictive power (solid lines) in representations from Deepset (yellow) and max-max GNN (red) greatly reduces the test MAPE. In contrast, the representation's predictive power for max-sum GNN barely reduces the model's test MAPE (tiny gap between dotted and solid purple lines).

Gamma distribution with zero-mean $\Gamma(2,\ 1/15) - 30$. We also consider more distributions with non-zero mean in Appendix A.1.

Findings. In Figure 3, while the max-sum GNN is robust to *zero-mean* additive label noise (dotted yellow and purple lines), its test error is much higher under non-zero-mean noise $\mathcal{N}(10,\ 15)$ (dotted red line) as the learned signal may be "shifted" by the non-centered label noise. Yet, max-sum GNNs' learned representations under these three types of label noise all predict the target function well when evaluating their predictive powers with 10% clean labels (solid lines in Figure 3).

Moreover, when we plot the representations (using PCA) from a max-sum GNN trained under 100% noise ratio with $\epsilon \sim \mathcal{N}(10,\ 15)$, the representations indeed correlate well with true labels (Figure 2). This explains why the representation learned under noisy labels can recover surprisingly good test performance despite that the original model has large test errors.

3.3 Instance-Dependent Label Noise

Realistic label noise is often instance-dependent. For example, an option is often incorrectly priced in the market, but its incorrect price (i.e., the noisy label) should depend on properties of the underlying stock. Such instance-dependent label noise is more challenging, as it may contain *spurious signals* that are easy to learn by certain architectures. In this section, we evaluate the representation' predictive power for three different GNNs trained with instance-dependent label noise.

Task and Label Noise. We experiment with a new task—computing the maximum node feature:

$$f(\mathcal{G}) := \max_{u \in \mathcal{G}} ||\boldsymbol{x}_u||_\infty. \tag{10}$$

To create a instance-dependent noise, we randomly replace the label with the maximum degree:

$$g(\mathcal{G}) := \max_{u \in \mathcal{G}} \sum_{v \in \mathcal{N}(u)} 1. \tag{11}$$

Architecture. We consider three GNNs: DeepSet [66], max-max GNN, and max-sum GNN. DeepSet can be interpreted as a special GNN that does not use neighborhood information:

$$h_{\mathcal{G}} = \mathrm{MLP}^{(1)}\left(\max_{u \in \mathcal{G}} \mathrm{MLP}^{(0)}\left(\boldsymbol{x}_u\right)\right). \tag{12}$$

Max-max GNN is a 2-layer GNN with max-aggregation and max-readout. Max-sum GNN is the same as the one in the previous section.

DeepSet and max-max GNN are well-aligned with the target function $f(\mathcal{G})$, as their MLP modules only need to learn simple linear functions. In contrast, max-sum GNN is more aligned with $g(\mathcal{G})$ than $f(\mathcal{G})$ since neither its MLP modules or sum-aggregation module can efficiently learn the max-operation in $f(\mathcal{G})$ [44, 46].

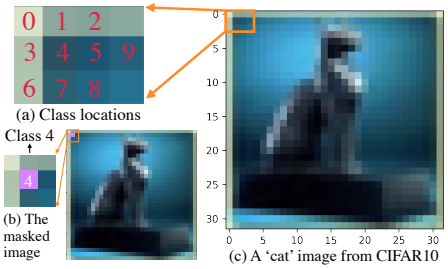

Figure 5: **Synthetic Labels on CIFAR-Easy.** For each image, we mask a pixel at the top left corner with pink color. Then the synthetic label for this image is the location of the pink pixel/mask (i.e., the cat image in the above example has label 4).

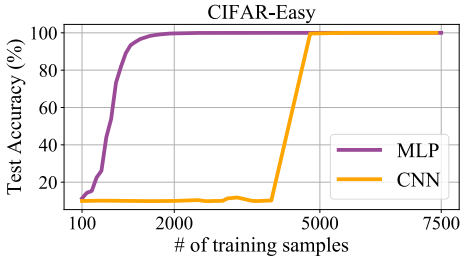

Figure 6: **Sample complexity of MLPs and CNNs on CIFAR-Easy.** Both MLPs and CNNs can achieve 100% test accuracy given sufficient training examples, but MLPs need far fewer examples than CNNs and thus are more sample-efficient on CIFAR-Easy.

Moreover, DeepSet cannot learn $g(\mathcal{G})$ as the model ignores *edge information*. We take the hidden representations before the last MLP modules in all three GNNs and compare their predictive power.

Findings. While all three GNNs have large test errors under high noise ratios (dotted lines in Figure 4), the predictive power in representations from GNNs more aligned with the target function — DeepSet (solid yellow line) and max-max GNN (solid red line) — significantly reduces the original models' test errors by 10 and 1000 times respectively. Yet, for the max-sum GNN, which is more aligned with the noise function, its representations' predictive power (solid purple line) barely decreases test error.

4 Experiments on Vision Datasets

Many noisy label training methods are benchmarked on image classification; thus, we also validate our hypothesis on image domains. We compare the representations' predictive power between MLPs and CNN-based networks using 10% clean labels. We further evaluate the predictive power in representations learned with SOTA methods. Predictive power on networks that aligned well with the target function could further improve SOTA method's test performance (Section 4.2). The final model also outperforms some sophisticated methods on noisy label training which also use clean labels (Appendix A.3). Experiment details are in Appendix B.3.

4.1 MLPs vs. CNN-based networks

To validate our hypothesis, we consider several target functions with different levels of alignments to MLPs and CNN-based networks. All models in this section are trained with standard procedures without any robust training methods or robust losses.

Datasets and Label Noise. We consider two types of target functions: one aligns better with CNN-based models than MLPs, and the other aligns better with MLPs than CNN-based networks.

1). CIFAR-10 and **CIFAR-100** [67] come with clean labels. Therefore, we generate two types of noisy labels following existing works: (1) **uniform label noise** randomly replaces the true labels with all possible labels, and (2) **flipped label noise** swaps the labels between similar classes (e.g., deer↔horse, dog↔cat) on CIFAR-10 [38], or flips the labels to the next class on CIFAR-100 [68].

2). CIFAR-Easy is a dataset modified on CIFAR-10 with labels generated by procedures in Figure 5 — the class/label of each image depends on the location of a special pixel. We consider three types of noisy labels on CIFAR-Easy: (1) **uniform label noise** and (2) **flipped label noise** (described as above); and (3) **instance-dependent label noise** which takes the original image classification label as the noisy label.

Architectures. On CIFAR-10/100, we evaluate the predictive power in representations for three architectures: 4-layer MLPs, 9-layer CNNs, and 18-layer PreAct ResNets [69]. On CIFAR-Easy, we compare between MLPs and CNNs. We take the representations before the penultimate layer when evaluating the predictive power for these networks.

As the designs of CNN-based networks (e.g., CNNs and ResNets) are similar to human perception system because of the receptive fields in convolutional layers and a hierarchical extraction of more and more abstracted features [70, 71], CNN-based networks are expected to *align better* with the target functions than MLPs on image classification datasets (e.g., CIFAR-10/100).

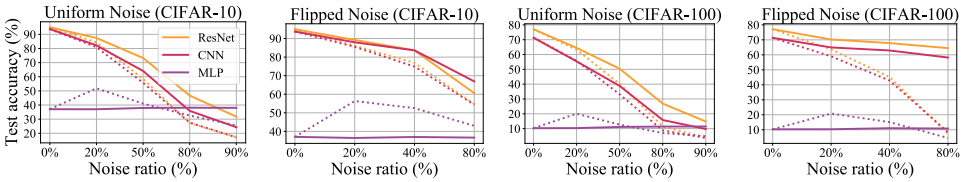

Figure 7: **CIFAR-10/100 with uniform and flipped label noise.** Each line indicates the raw test accuracy (dotted) and the predictive power in representations (solid) learned by a model trained across various noise ratios. As CNN-based networks align better with image classification tasks than MLPs, their representations' predictive power (solid yellow and red lines) are higher than that of MLPs (solid purple lines) on most noise ratios.

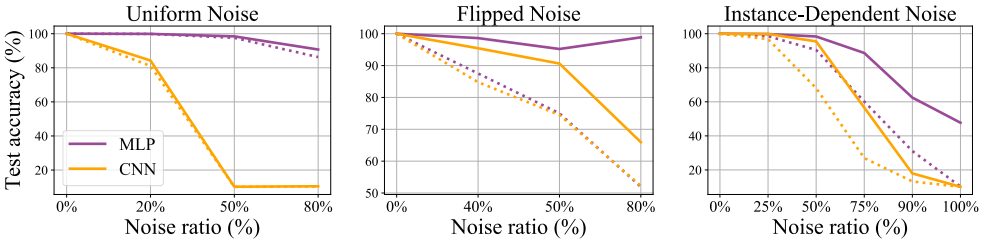

Figure 8: **CIFAR-Easy with uniform, flipped, and instance-dependent label noise.** Each line indicates the raw test accuracy (dotted) and the predictive power in representations (solid) learned by a model trained across various noise ratios. As MLPs align better with the target function than CNN-based networks on CIFAR-Easy, their representations' predictive power on MLPs (solid purple lines) are consistently better than that of CNNs (solid yellow lines) across various noise ratios and noise types.

On the other hand, on CIFAR-Easy, while both CNNs and MLPs can generalize perfectly given sufficient training examples, MLPs have a much smaller sample complexity than CNNs (Figure 6). Thus, both MLP and CNN are *well-aligned* with the target function on CIFAR-Easy, but MLP is *better-aligned* than CNN according to Theorem 2. Moreover, since the instance-dependent label on CIFAR-Easy is the original image classification label, CNN is also *aligned* with this instance-dependent noise function on CIFAR-Easy.

Experimental Results. First, we empirically verify our hypothesis that *networks better-aligned with the target function have more predictive representations*. As expected, across most noise ratios on CIFAR-10/100, the representations in CNN-based networks (i.e., CNN and ResNet) are more predictive than those in MLPs (Figure 7) under both types of label noise. Moreover, the predictive power in representations learned by less aligned networks (i.e., MLPs) sometimes are even worse than the vanilla-trained models' test performance, suggesting that the noisy representations on less aligned networks may be more corrupted and less linearly separable. On the other hand, across all three types of label noise on CIFAR-Easy, MLPs, which align better with the target function, have more predictive representations than CNNs (Figure 8).

We also observe that *models with similar test performance could have various levels of predictive powers in their learned representations*. For example, in Figure 8, while the test accuracies of MLPs and CNNs are very similar on CIFAR-Easy under flipped label noise (i.e., dotted purple and yellow lines overlap), the predictive power in representations from MLPs is much stronger than the one from CNNs (i.e., solid purple lines are much higher than yellow lines). This also suggests that when trained with noisy labels, if we do not know which architecture is more aligned with the underlying target function, we can evaluate the predictive power in their representations to test alignment.

Table 1: **Comparison of different networks' test accuracies (%) on CIFAR-10/100.** We color a test accuracy in red if it is lower than the test accuracy from vanilla training, and we color it in green if it is higher.

Model	Setting	CIFAR10							CIFAR100						
		Uniform noise				Flipped noise			Uniform noise				Flipped noise		
		20%	50%	80%	90%	20%	40%	80%	20%	50%	80%	90%	20%	40%	80%
4-layer FC (MLP)	Vanilla training	51.8	41.0	32.5	25.6	56.4	52.5	43.0	20.1	12.7	7.0	4.9	20.8	15.1	4.5
	DivideMix [38]	62.2	55.2	34.4	28.1	60.2	56.8	44.0	32.8	28.0	13.9	7.2	31.5	22.3	1.3
	DivideMix's Predictive Power	38.6	38.6	38.8	38.2	38.5	39.0	38.8	11.1	11.8	12.4	11.6	11.1	12.0	11.9
PreAct ResNet18	Vanilla training	84.4	58.5	27.3	17.2	86.1	76.9	54.7	63.2	40.2	11.5	3.9	63.6	45.2	7.4
	DivideMix [38]	95.7	94.4	92.9	75.4	94.0	92.1	56.2	76.9	74.2	59.6	31.0	77.0	55.2	0.2
	DivideMix's Predictive Power	96.0	94.8	93.5	83.8	94.9	94.0	93.6	76.6	73.9	60.9	39.3	76.8	74.8	76.1
9-layer CNN	Vanilla training	80.9	55.7	27.5	17.1	85.5	74.9	54.4	55.8	33.1	8.8	3.7	59.0	42.8	8.3
	DivideMix [38]	94.5	93.4	91.2	78.2	92.9	89.8	55.3	71.4	69.0	51.8	22.9	71.3	53.0	0.3
	DivideMix's Predictive Power	94.5	93.6	91.4	81.8	93.6	92.1	90.1	69.9	67.1	50.4	26.3	70.2	69.0	68.8

Table 2: **Comparison of different networks' test accuracies (%) on CIFAR-Easy**. We color a test accuracy in red if it is lower than the test accuracy from vanilla training, and we color it in green if it is higher.

Model	Setting	Uniform noise				Flipped noise			Spurious noise				
		0%	50%	80%	90%	40%	50%	80%	25%	50%	75%	90%	100%
4-layer FC (MLP)	Vanilla training	100.00	99.88	97.57	86.29	87.52	75.01	51.94	98.55	90.46	60.16	31.08	10.25
	DivideMix [38]	98.35	10.00	99.99	16.22	100.00	88.36	50.04	100.00	100.00	45.59	14.16	10.10
	DivideMix's Predictive Power	100.00	100.00	100.00	99.94	100.00	100.00	100.00	100.00	100.00	100.00	98.66	99.65
9-layer CNN	Vanilla training	100.00	81.08	10.15	10.36	84.80	74.50	52.24	96.84	68.07	26.97	13.24	10.07
	DivideMix [38]	100.00	100.00	100.00	10.00	92.75	86.33	50.60	99.96	99.82	10.45	10.09	10.14
	DivideMix's Predictive Power	100.00	100.00	100.00	10.09	99.33	98.42	96.76	100.00	99.99	14.99	10.70	10.15

We further discover that *for networks well-aligned with the target function, its learned representations are more predictive when the noise function shares more mutual information with the target function*. We compute the empirical mutual information between the noisy training labels and the original clean labels across different noise ratios on various types of label noise. The predictive power in representations improves as the mutual information increases (Figure 11 in Appendix A). This explains why the predictive power for a network is often higher under flipped noise than uniform noise: at the same noise ratio, flipped noise has higher mutual information than uniform noise. Moreover, comparing across the three datasets in Figure 11, we observe the growth rate of a network's predictive power w.r.t. the mutual information depends on both the intrinsic difficulties of the learning task and the alignment between the network and the target function.

4.2 Predictive Power in Representations for Models Trained with SOTA Methods

As previous experiments are on standard training procedures, we also validate our hypothesis on models learned with SOTA methods on noisy label training. We evaluate the representations' predictive power for models trained with the SOTA method, DivideMix [38], which leverages techniques from semi-supervised learning to treat examples with unreliable labels as unlabeled data.

We compare (1) the test performance for models trained with standard procedures on noisy labels (denoted as **Vanilla training**), (2) the SOTA method's test performance (denoted as **DivideMix**), and (3) the predictive power in representations from models trained with DivideMix in (2) (denoted as **DivideMix's Predictive Power**).

We discover that *the effectiveness of DivideMix also depends on the alignment between the network and the target/noise functions*. DivideMix only slightly improves the test accuracy of MLPs on CIFAR-10/100 (Table 1), and DivideMix's predictive power does not improve the test performance of MLPs, either. In Table 2, DivideMix also barely helps CNNs as they are well-aligned with the instance-dependent noise, where the noisy label is the original image classification label.

Moreover, we observe that *even for networks well-aligned with the target function, DivideMix may only slightly improve or do not improve its test performance at all* (e.g., red entries of DivideMix on MLPs in Table 2). Yet, the representations learned with DivideMix can still be very predictive: the predictive power can achieve over 50% improvements over DivideMix for CNN-based models on CIFAR-10/100 (e.g., 80% flipped noise), and the improvements can be over 80% for MLPs on CIFAR-Easy (e.g., 90% uniform noise).

Tables 1 and 2 shows that the representations' predictive power on networks well aligned with the target function could further improve SOTA test performance. Appendix A.3 further demonstrates that on large-scale datasets with real-world noisy labels, the predictive power in well-aligned networks could outperform sophisticated methods that also use clean labels (Table 7 and Table 8).

5 Concluding Remarks

This paper is an initial step towards formally understanding how a network's architectures impacts its robustness to noisy labels. We formalize our intuitions and hypothesize that a network better-aligned with the target function would learn more predictive representations under noisy label training. We prove our hypothesis on a simplified noisy setting and conduct systematic experiments across various noisy settings to further validate our hypothesis.

Our empirical results along with Theorem 2 suggest that knowing more structures of the target function can help design more robust architectures. In practice, although an exact mathematical formula for a decomposition of a given target function is often hard to obtain, a high-level decomposition of the target function often exists for real-world tasks and will be helpful in designing robust architectures — a direction undervalued by existing works on learning with noisy labels.

References

[1] Rion Snow, Brendan O'connor, Dan Jurafsky, and Andrew Y Ng. Cheap and fast–but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of Empirical Methods in Natural Language Processing*, 2008.

[2] Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*, pages 2424–2432, 2010.

[3] Yan Yan, Rómer Rosales, Glenn Fung, Ramanathan Subramanian, and Jennifer Dy. Learning from multiple annotators with varying expertise. *Machine learning*, 95(3):291–327, 2014.

[4] Xiyu Yu, Tongliang Liu, Mingming Gong, and Dacheng Tao. Learning with biased complementary labels. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 68–83, 2018.

[5] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

[6] Lu Jiang, Di Huang, Mason Liu, and Weilong Yang. Beyond synthetic noise: Deep learning on controlled noisy labels. In *ICML*, 2020. URL https://arxiv.org/abs/1911.09781.

[7] Wei Liu, Yu-Gang Jiang, Jiebo Luo, and Shih-Fu Chang. Noise resistant graph ranking for improved web image search. In *CVPR 2011*, pages 849–856. IEEE, 2011.

[8] Aritra Ghosh, Himanshu Kumar, and PS Sastry. Robust loss functions under label noise for deep neural networks. *arXiv preprint arXiv:1712.09482*, 2017.

[9] Yueming Lyu and Ivor W Tsang. Curriculum loss: Robust learning and generalization against label corruption. *arXiv preprint arXiv:1905.10045*, 2019.

[10] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 322–330, 2019.

[11] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in neural information processing systems*, pages 8778–8788, 2018.

[12] Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. *arXiv preprint arXiv:1904.11238*, 2019.

[13] Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. Active bias: Training more accurate neural networks by emphasizing high variance samples. In *Advances in Neural Information Processing Systems*, pages 1002–1012, 2017.

[14] Bo Han, Gang Niu, Xingrui Yu, Quanming Yao, Miao Xu, Ivor W Tsang, and Masashi Sugiyama. Sigua: Forgetting may make learning with noisy labels more robust. In *Proceedings of the 37-th International conference on machine learning (ICML-20)*, 2020.

[15] Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. In *Advances in neural information processing systems*, pages 10456–10465, 2018.

[16] Xingjun Ma, Yisen Wang, Michael E Houle, Shuo Zhou, Sarah M Erfani, Shu-Tao Xia, Sudanthi Wijewickrema, and James Bailey. Dimensionality-driven learning with noisy labels. *arXiv preprint arXiv:1806.02612*, 2018.

[17] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952, 2017.

[18] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

[19] Hwanjun Song, Minseok Kim, and Jae-Gil Lee. Selfie: Refurbishing unclean samples for robust deep learning. In *International Conference on Machine Learning*, pages 5907–5915, 2019.

[20] Ruxin Wang, Tongliang Liu, and Dacheng Tao. Multiclass learning with partially corrupted labels. *IEEE transactions on neural networks and learning systems*, 29(6):2568–2580, 2017.

[21] Pengfei Chen, Benben Liao, Guangyong Chen, and Shengyu Zhang. Understanding and utilizing deep neural networks trained with noisy labels. *arXiv preprint arXiv:1905.05040*, 2019.

[22] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in neural information processing systems*, pages 8527–8537, 2018.

[23] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313, 2018.

[24] Eran Malach and Shai Shalev-Shwartz. Decoupling" when to update" from" how to update". In *Advances in Neural Information Processing Systems*, pages 960–970, 2017.

[25] Duc Tam Nguyen, Chaithanya Kumar Mummadi, Thi Phuong Nhung Ngo, Thi Hoai Phuong Nguyen, Laura Beggel, and Thomas Brox. Self: Learning to filter noisy labels with self-ensembling. *arXiv preprint arXiv:1910.01842*, 2019.

[26] Yanyao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In *International Conference on Machine Learning*, pages 5739–5748. PMLR, 2019.

[27] Yisen Wang, Weiyang Liu, Xingjun Ma, James Bailey, Hongyuan Zha, Le Song, and Shu-Tao Xia. Iterative learning with open-set noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8688–8696, 2018.

[28] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor W Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? *arXiv preprint arXiv:1901.04215*, 2019.

[29] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[30] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. *arXiv preprint arXiv:1901.09960*, 2019.

[31] Simon Jenni and Paolo Favaro. Deep bilevel learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 618–633, 2018.

[32] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

[33] Ryutaro Tanno, Ardavan Saeedi, Swami Sankaranarayanan, Daniel C Alexander, and Nathan Silberman. Learning from noisy labels by regularized estimation of annotator confusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11244–11253, 2019.

[34] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[35] Luís PF Garcia, André CPLF de Carvalho, and Ana C Lorena. Noise detection in the meta-learning level. *Neurocomputing*, 176:14–25, 2016.

[36] Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli. Learning to learn from noisy labeled data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5051–5059, 2019.

[37] Yifan Ding, Liqiang Wang, Deliang Fan, and Boqing Gong. A semi-supervised two-stage approach to learning from noisy labels. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1215–1224. IEEE, 2018.

[38] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020.

[39] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *arXiv preprint arXiv:2007.00151*, 2020.

[40] Yan Yan, Zhongwen Xu, Ivor W Tsang, Guodong Long, and Yi Yang. Robust semi-supervised learning through label aggregation. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[41] Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. *arXiv preprint arXiv:1706.05394*, 2017.

[42] Hartmut Maennel, Ibrahim Alabdulmohsin, Ilya Tolstikhin, Robert JN Baldock, Olivier Bousquet, Sylvain Gelly, and Daniel Keysers. What do neural networks learn when trained with random labels? *arXiv preprint arXiv:2006.10455*, 2020.

[43] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

[44] Keyulu Xu, Jingling Li, Mozhi Zhang, Simon S. Du, Ken ichi Kawarabayashi, and Stefanie Jegelka. What can neural networks reason about? In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rJxbJeHFPS.

[45] Simon S Du, Kangcheng Hou, Russ R Salakhutdinov, Barnabas Poczos, Ruosong Wang, and Keyulu Xu. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. In *Advances in Neural Information Processing Systems*, pages 5724–5734, 2019.

[46] Keyulu Xu, Mozhi Zhang, Jingling Li, Simon S Du, Ken-ichi Kawarabayashi, and Stefanie Jegelka. How neural networks extrapolate: From feedforward to graph neural networks. *arXiv preprint arXiv:2009.11848*, 2020.

[47] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2691–2699, 2015.

[48] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

[49] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *arXiv preprint arXiv:2008.03703*, 2020.

[50] Kimin Lee, Sukmin Yun, Kibok Lee, Honglak Lee, Bo Li, and Jinwoo Shin. Robust inference via generative classifiers for handling noisy labels. In *International Conference on Machine Learning*, pages 3763–3772. PMLR, 2019.

[51] Dara Bahri, Heinrich Jiang, and Maya Gupta. Deep k-nn for noisy labels. In *International Conference on Machine Learning*, pages 540–550. PMLR, 2020.

[52] Pengxiang Wu, Songzhu Zheng, Mayank Goswami, Dimitris Metaxas, and Chao Chen. A topological filter for learning with label noise. *arXiv preprint arXiv:2012.04835*, 2020.

[53] Katherine L Hermann and Andrew K Lampinen. What shapes feature representations? exploring datasets, architectures, and training. *arXiv preprint arXiv:2006.12433*, 2020.

[54] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. *arXiv preprint arXiv:2006.07710*, 2020.

[55] Amartya Sanyal, Puneet K Dokania, Varun Kanade, and Philip HS Torr. How benign is benign overfitting? *arXiv preprint arXiv:2007.04028*, 2020.

[56] Katherine L Hermann, Ting Chen, and Simon Kornblith. The origins and prevalence of texture bias in convolutional neural networks. *arXiv preprint arXiv:1911.09071*, 2019.

[57] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[58] Wei Hu, Zhiyuan Li, and Dingli Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Hke3gyHYwH.

[59] Hao Cheng, Zhaowei Zhu, Xingyu Li, Yifei Gong, Xing Sun, and Yang Liu. Learning with instance-dependent label noise: A sample sieve approach. *arXiv preprint arXiv:2010.02347*, 2020.

[60] Xinshao Wang, Yang Hua, Elyor Kodirov, and Neil M Robertson. Proselflc: Progressive self label correction for target revising in label noise. *arXiv preprint arXiv:2005.03788*, 2020.

[61] Ryoma Sato, Makoto Yamada, and Hisashi Kashima. Approximation ratios of graph neural networks for combinatorial problems. In *Advances in Neural Information Processing Systems*, pages 4081–4090, 2019.

[62] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[63] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[64] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1273–1272, 2017.

[65] Denny Wu and Ji Xu. On the optimal weighted ℓ_2 regularization in overparameterized linear regression. *arXiv preprint arXiv:2006.05800*, 2020.

[66] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J Smola. Deep sets. corr abs/1703.06114 (2017). *arXiv preprint arXiv:1703.06114*, 2017.

[67] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[68] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pages 1196–1204, 2013.

[69] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[70] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[71] Saeed Reza Kheradpisheh, Masoud Ghodrati, Mohammad Ganjtabesh, and Timothée Masquelier. Deep networks can resemble human feed-forward vision in invariant object recognition. *Scientific reports*, 6(1): 1–24, 2016.

[72] Zizhao Zhang, Han Zhang, Sercan O Arik, Honglak Lee, and Tomas Pfister. Distilling effective supervision from severe label noise. In *CVPR 2020*, pages 9294–9303. IEEE, 2020.

[73] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*, 2018.

[74] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems*, pages 1919–1930, 2019.

[75] Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5447–5456, 2018.

[76] Jiangfan Han, Ping Luo, and Xiaogang Wang. Deep self-learning from noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5138–5147, 2019.

[77] Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

[78] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[79] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016.

[80] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.

[81] David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *Special Interest Group on Information Retrieval*, 1994.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See supplemental material.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix B.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We report error bars as shaded area in our plots.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix B.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] .
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]