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

Jingling Li [1] **Mozhi Zhang** [1] **Keyulu Xu** [2] **John Dickerson** [1] **Jimmy Ba** [3]

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 or online queries, which are inevitably noisy. To learn from these noisy labels, previous works propose many techniques, including designing robust losses (Ghosh et al., 2017; Lyu & Tsang, 2019), adjusting loss before gradient updates (Reed et al., 2014; Hendrycks et al., 2018; Arazo et al., 2019), selecting trust-worthy samples (Han et al., 2018; Jiang et al., 2018), applying robust regularization in training (Goodfellow et al., 2015; Hendrycks et al., 2019), meta-learning to avoid over-fitting (Garcia et al., 2016; Li et al., 2019), and semi-supervised learning (Yan et al., 2016; Li et al., 2020) 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 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 (Arpit et al., 2017; Maennel et al., 2020). 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 representations with a small set of clean labels and evaluate the linear model's test error (Alain & Bengio, 2016).

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 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 2.2, adapted from (Xu et al., 2020a).

Our proposed framework provides initial theoretical support for our findings on a simplified noisy setting (Theorem 1). Empirically, we validate our findings on synthetic graph algorithmic tasks (Section 3) by designing several variants of Graph Neural Networks (GNNs), whose theoretical properties and alignment with algorithmic functions have been

[1]Department of Computer Science, University of Maryland, College Park, MD, USA [2]Electrical Engineering Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA [3]Department of Computer Science, Machine Learning Group, University of Toronto, Toronto, Canada. Correspondence to: Jingling Li <jingling@cs.umd.edu>.

Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning.,

well-studied (Du et al., 2019; Xu et al., 2020a;b). Many noisy label training methods are applied to image classification datasets, so we also validate our findings on image domains using different architectures (Appendix C).

Most of our analysis and experiments use standard neural network training. Interestingly, we find similar results when using DivideMix (Li et al., 2020), 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. Moreover, for aligned networks, the predictive power could further improve the test performance of SOTA methods or sophisticated methods that use clean labels (Appendix E.3), especially on class-dependent or instance-dependent label noise where current noisy-label-training methods are less effective.

2. Overview of Our Framework

In this section, we give formal definitions for "predictive power" and "alignment," and present our main hypothesis as well as our main theorem. Due to space limit, our problem setting is introduced in Appendix B.

2.1. Predictive Power in Representations

A network's robustness is often measured by its test performance after trained 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

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

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 by applying linear regression to obtain a trained linear model L, which avoids the issue of local minima as it is a convex problem; then we evaluate L on the test set.

2.2. 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, which Xu et al. (2020a) define as a sample complexity measure in a PAC learning framework.

Definition 2. *(Alignment, simplified based on Xu et al. (2020a)).* 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

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

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 (Xu et al., 2020a), we use *alignment* more as a qualitative rather than quantitative measure in this paper.

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.

2.3. Better Alignment Implies Better Robustness

Xu et al. (2020a) prove that better alignment implies better sample complexity and vice versa (Theorem 2 in Appendix F), and we hypothesize that a network better-aligned with the target function (smaller *Alignment*$(\mathcal{N}, f, \epsilon, \delta)$) would learn more predictive representations (smaller $P_j(f, \mathcal{N}, \mathcal{C})$) 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,

$$\textit{Alignment}(\mathcal{N}, f, \epsilon, \delta) \downarrow \implies P_j(f, \mathcal{N}, \mathcal{C}) \downarrow, \qquad (3)$$

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 F for a full statement of our main theorem with detailed assumptions.

Theorem 1. *(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 for $\mathcal{N}$ trained 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 $\mathcal{X}$ 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 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 (Xu et al., 2020a; Du et al., 2019; Xu et al., 2020b; Sato et al., 2019). 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 D.2. All training details are in Appendix E.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 (Battaglia et al., 2018; Scarselli et al., 2009).

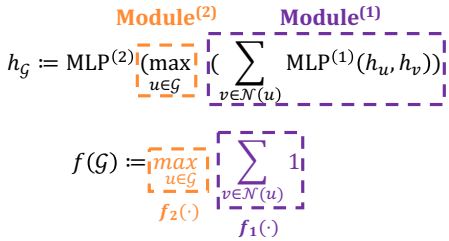


Figure 1. **Max-sum GNN is well-aligned with the task maximum degree.** The max-sum GNN can be decomposed into two modules: Module$^{(1)}$ and Module$^{(1)}$, and the target function can be write as $f(\mathcal{G}) = f_1(f_1(\mathcal{G}))$. Note that $f_1(\cdot)$ can be easily learned by Module$^{(1)}$,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

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 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 (Gilmer et al., 2017). We can optionally compute a graph representation $\boldsymbol{h}_{\mathcal{G}}$ by aggregating the final node representations with another MLP module. Formally, (1) $\boldsymbol{h}_u^{(k)} := \sum_{v \in \mathcal{N}(u)} \text{MLP}^{(k)}(\boldsymbol{h}_u^{(k-1)}, \boldsymbol{h}_v^{(k-1)})$; (2) $\boldsymbol{h}_{\mathcal{G}} := \text{MLP}^{(K+1)}(\sum_{u \in \mathcal{G}} \boldsymbol{h}_u^{(K)})$.

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

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: $f(\mathcal{G}) := \max_{u \in \mathcal{G}} \sum_{v \in \mathcal{N}(u)} 1$. 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): (1) $\boldsymbol{h}_{\mathcal{G}} := \text{MLP}^{(2)}\left(\max_{u \in \mathcal{G}} \sum_{v \in \mathcal{N}(u)} \text{MLP}^{(1)}\left(\boldsymbol{h}_u, \boldsymbol{h}_v\right)\right)$; (2) $\boldsymbol{h}_u := \sum_{v \in \mathcal{N}(u)} \text{MLP}^{(0)}\left(\boldsymbol{x}_u, \boldsymbol{x}_v\right)$.

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

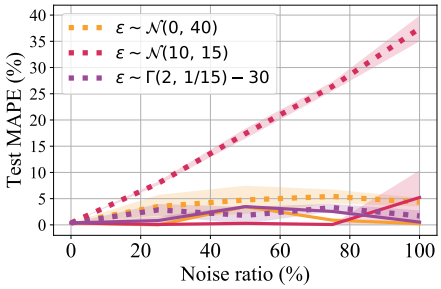


Figure 2. **Representations are very predictive for a GNN well-aligned with the target function under additive label noise.** On the maximum degree task, the representations' predictive power (solid lines) achieves low test MAPE ($< 5\%$) across all three types of noise for the max-sum GNN, despite that 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.

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

Findings. In Figure 2, while max-sum GNN is robust to *zero-mean* additive label noise (dotted yellow and purple lines), its test error is much higher on 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 on these three types of label noise all predict the target function well using 10% clean labels (solid lines in Figure 2).

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 8 in Appendix D.1). 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. Here, 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: $f(\mathcal{G}) := \max_{u\in\mathcal{G}} ||\boldsymbol{x}_u||_\infty$. To create a instance-dependent noise, we randomly replace the label with the maximum degree of the graph: $g(\mathcal{G}) := \max_{u\in\mathcal{G}} \sum_{v\in\mathcal{N}(u)} 1$.

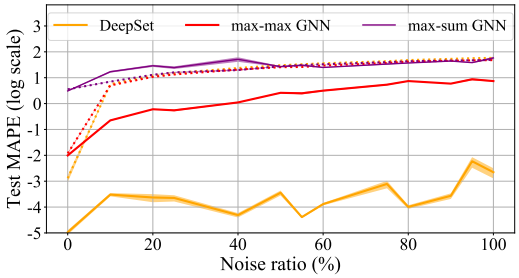


Figure 3. **Representations are more predictive for GNNs more aligned with the target function, and less predictive for GNNs more aligned with the noise function**. On the maximum node feature task, while all three GNNs have large test errors under high noise ratios (dotted lines), the predictive power (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).

Architecture. We consider three GNNs: DeepSet (Zaheer et al., 2017), max-max GNN, and max-sum GNN. DeepSet can be interpreted as a special GNN that does not use neighborhood information: $h_\mathcal{G} = \text{MLP}^{(1)}\left(\max_{u\in\mathcal{G}}\text{MLP}^{(0)}\left(\boldsymbol{x}_u\right)\right)$. 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 $f(\mathcal{G})$ (Xu et al., 2020a;b).

Moreover, DeepSet cannot learn $g(\mathcal{G})$ as it ignores *edge information*. We take the hidden representations before the last MLP module to evaluate the GNNs' predictive power.

Findings. While all three GNNs have large test errors under high noise ratios (dotted lines in Figure 3), 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. Concluding Remarks

Our results suggest that knowing more structures of the target function can help design more robust architectures, which is also a direction undervalued by existing works on learning with noisy labels.

References

Alain, G. and Bengio, Y. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Arazo, E., Ortego, D., Albert, P., O'Connor, N. E., and McGuinness, K. Unsupervised label noise modeling and loss correction. *arXiv preprint arXiv:1904.11238*, 2019.

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. A closer look at memorization in deep networks. *arXiv preprint arXiv:1706.05394*, 2017.

Bahri, D., Jiang, H., and Gupta, M. Deep k-nn for noisy labels. In *International Conference on Machine Learning*, pp. 540–550. PMLR, 2020.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Cheng, H., Zhu, Z., Li, X., Gong, Y., Sun, X., and Liu, Y. Learning with instance-dependent label noise: A sample sieve approach. *arXiv preprint arXiv:2010.02347*, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Du, S. S., Hou, K., Salakhutdinov, R. R., Poczos, B., Wang, R., and Xu, K. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. In *Advances in Neural Information Processing Systems*, pp. 5724–5734, 2019.

Feldman, V. and Zhang, C. What neural networks memorize and why: Discovering the long tail via influence estimation. *arXiv preprint arXiv:2008.03703*, 2020.

Garcia, L. P., de Carvalho, A. C., and Lorena, A. C. Noise detection in the meta-learning level. *Neurocomputing*, 176:14–25, 2016.

Ghosh, A., Kumar, H., and Sastry, P. Robust loss functions under label noise for deep neural networks. *arXiv preprint arXiv:1712.09482*, 2017.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pp. 1273–1272, 2017.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., and Sugiyama, M. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in neural information processing systems*, pp. 8527–8537, 2018.

Han, J., Luo, P., and Wang, X. Deep self-learning from noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5138–5147, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016.

Hendrycks, D., Mazeika, M., Wilson, D., and Gimpel, K. Using trusted data to train deep networks on labels corrupted by severe noise. In *Advances in neural information processing systems*, pp. 10456–10465, 2018.

Hendrycks, D., Lee, K., and Mazeika, M. Using pre-training can improve model robustness and uncertainty. *arXiv preprint arXiv:1901.09960*, 2019.

Hermann, K. L. and Lampinen, A. K. What shapes feature representations? exploring datasets, architectures, and training. *arXiv preprint arXiv:2006.12433*, 2020.

Hermann, K. L., Chen, T., and Kornblith, S. The origins and prevalence of texture bias in convolutional neural networks. *arXiv preprint arXiv:1911.09071*, 2019.

Hu, W., Li, Z., and Yu, D. 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.

Jiang, L., Zhou, Z., Leung, T., Li, L.-J., and Fei-Fei, L. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pp. 2304–2313, 2018.

Kheradpisheh, S. R., Ghodrati, M., Ganjtabesh, M., and Masquelier, T. Deep networks can resemble human feed-forward vision in invariant object recognition. *Scientific reports*, 6(1):1–24, 2016.

Krizhevsky, A. Learning multiple layers of features from tiny images. 2009.

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

Lee, K., Yun, S., Lee, K., Lee, H., Li, B., and Shin, J. Robust inference via generative classifiers for handling noisy labels. In *International Conference on Machine Learning*, pp. 3763–3772. PMLR, 2019.

Lee, K.-H., He, X., Zhang, L., and Yang, L. Cleannet: Transfer learning for scalable image classifier training with label noise. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5447–5456, 2018.

Lewis, D. D. and Gale, W. A. A sequential algorithm for training text classifiers. In *Special Interest Group on Information Retrieval*, 1994.

Li, J., Wong, Y., Zhao, Q., and Kankanhalli, M. S. Learning to learn from noisy labeled data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5051–5059, 2019.

Li, J., Socher, R., and Hoi, S. C. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020.

Li, W., Wang, L., Li, W., Agustsson, E., and Van Gool, L. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

Liu, S., Niles-Weed, J., Razavian, N., and Fernandez-Granda, C. Early-learning regularization prevents memorization of noisy labels. *arXiv preprint arXiv:2007.00151*, 2020.

Lyu, Y. and Tsang, I. W. Curriculum loss: Robust learning and generalization against label corruption. *arXiv preprint arXiv:1905.10045*, 2019.

Maennel, H., Alabdulmohsin, I., Tolstikhin, I., Baldock, R. J., Bousquet, O., Gelly, S., and Keysers, D. What do neural networks learn when trained with random labels? *arXiv preprint arXiv:2006.10455*, 2020.

Miyato, T., Maeda, S.-i., Koyama, M., and Ishii, S. 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.

Montavon, G., Samek, W., and Müller, K.-R. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

Natarajan, N., Dhillon, I. S., Ravikumar, P. K., and Tewari, A. Learning with noisy labels. In *Advances in neural information processing systems*, pp. 1196–1204, 2013.

Nguyen, D. T., Mummadi, C. K., Ngo, T. P. N., Nguyen, T. H. P., Beggel, L., and Brox, T. Self: Learning to filter noisy labels with self-ensembling. *arXiv preprint arXiv:1910.01842*, 2019.

Patrini, G., Rozza, A., Krishna Menon, A., Nock, R., and Qu, L. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1944–1952, 2017.

Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., and Rabinovich, A. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

Ren, M., Zeng, W., Yang, B., and Urtasun, R. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*, 2018.

Sanyal, A., Dokania, P. K., Kanade, V., and Torr, P. H. How benign is benign overfitting? *arXiv preprint arXiv:2007.04028*, 2020.

Sato, R., Yamada, M., and Kashima, H. Approximation ratios of graph neural networks for combinatorial problems. In *Advances in Neural Information Processing Systems*, pp. 4081–4090, 2019.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

Shah, H., Tamuly, K., Raghunathan, A., Jain, P., and Netrapalli, P. The pitfalls of simplicity bias in neural networks. *arXiv preprint arXiv:2006.07710*, 2020.

Shu, J., Xie, Q., Yi, L., Zhao, Q., Zhou, S., Xu, Z., and Meng, D. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems*, pp. 1919–1930, 2019.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016.

Wang, X., Hua, Y., Kodirov, E., and Robertson, N. M. Proselflc: Progressive self label correction for target revising in label noise. *arXiv preprint arXiv:2005.03788*, 2020.

Wu, P., Zheng, S., Goswami, M., Metaxas, D., and Chen, C. A topological filter for learning with label noise. *arXiv preprint arXiv:2012.04835*, 2020.

Xiao, T., Xia, T., Yang, Y., Huang, C., and Wang, X. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2691–2699, 2015.

Xu, K., Li, J., Zhang, M., Du, S. S., ichi Kawarabayashi, K., and Jegelka, S. What can neural networks reason about? In *International Conference on Learning Representations*, 2020a. URL https://openreview.net/forum?id=rJxbJeHFPS.

Xu, K., Zhang, M., Li, J., Du, S. S., Kawarabayashi, K.-i., and Jegelka, S. How neural networks extrapolate: From feedforward to graph neural networks. *arXiv preprint arXiv:2009.11848*, 2020b.

Yan, Y., Xu, Z., Tsang, I. W., Long, G., and Yang, Y. Robust semi-supervised learning through label aggregation. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Póczos, B., Salakhutdinov, R., and Smola, A. J. Deep sets. corr abs/1703.06114 (2017). *arXiv preprint arXiv:1703.06114*, 2017.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

Zhang, Z., Zhang, H., Arik, S. O., Lee, H., and Pfister, T. Distilling effective supervision from severe label noise. In *CVPR 2020*, pp. 9294–9303. IEEE, 2020.

A. 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 (Zhang et al., 2017), it has been observed that networks learn simple patterns first (Arpit et al., 2017), converge faster on downstream tasks (Maennel et al., 2020), and benefit from memorizing atypical training samples (Feldman & Zhang, 2020).

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 (Lyu & Tsang, 2019; Han et al., 2018; Jiang et al., 2018; Li et al., 2020; Liu et al., 2020) and learn useful feature patterns (Hendrycks et al., 2018; Lee et al., 2019; Bahri et al., 2020; Wu et al., 2020). 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 (Arpit et al., 2017; Hermann & Lampinen, 2020; Shah et al., 2020; Sanyal et al., 2020), and how to interpret the learned representations on clean data (Alain & Bengio, 2016; Hermann & Lampinen, 2020; Hermann et al., 2019; Montavon et al., 2018). Our paper focuses more on the predictive power rather than the explanatory power in the learned representations. We adapt the method in (Alain & Bengio, 2016) 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 (Arpit et al., 2017; Hermann & Lampinen, 2020; Shah et al., 2020; Sanyal et al., 2020). Our work formalizes what is considered as a simple pattern for a given network via architectural alignments, and we extend the definition of alignment in (Xu et al., 2020a) to noisy settings.

B. 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 of clean labels, and $\mathcal{I}'$ contains indices of 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. For a label space $\mathcal{Y} \subseteq \mathbb{R}$, we consider two types of label noise: a) **additive label noise** (Hu et al., 2020): $\hat{y} := y + \epsilon$, 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 $\boldsymbol{x}$.

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. Previous works on noisy label learning commonly study uniform and flipped label noise. A few recent works (Cheng et al., 2020; Wang et al., 2020) explore the instance-dependent label noise as it is more realistic.

C. 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 C.2). The final model also outperforms some sophisticated methods on noisy label training which also use clean labels (Appendix D.3). Experiment details are in Appendix E.3.

C.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** (Krizhevsky, 2009) 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 (Li et al., 2020), or flips the labels to the next class on CIFAR-100 (Natarajan et al., 2013).

2). CIFAR-Easy is a dataset modified on CIFAR-10 with labels generated by procedures in Figure 4 — 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.

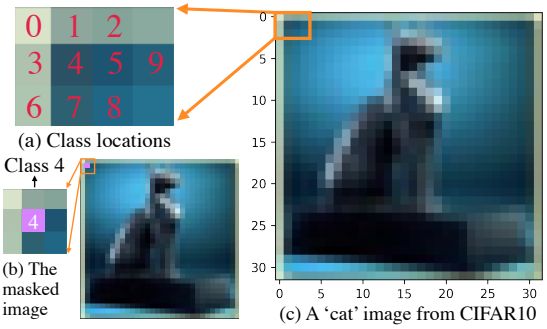


Figure 4. **Synthetic Labels on CIFAR-Easy.** For each image, we mask a pixel at the top left corner with pink color. Then the synthetic label for this image is the location of the pink pixel/mask (i.e., the cat image in the above example has Class 4).

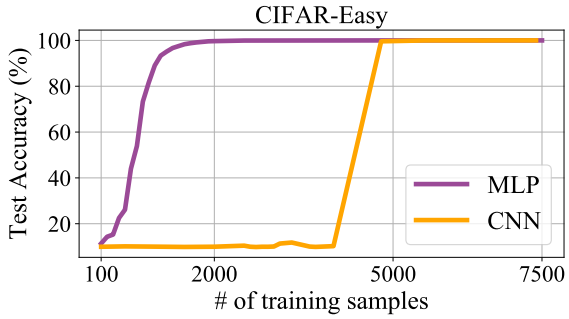


Figure 5. **Sample complexity of MLPs and CNNs on CIFAR-Easy.** Both MLPs and CNNs can achieve 100% test accuracy given sufficient samples, but MLPs need far fewer training data than CNNs and thus are more sample-efficient on CIFAR-Easy.

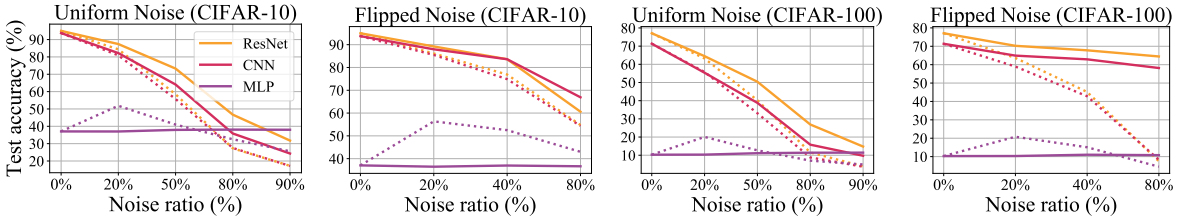


Figure 6. **CIFAR-10/100 with uniform and flipped label noise.** Each line indicates the raw test accuracy (dotted) and the predictive power in representations (solid) learned by a model trained across various noise ratios. As CNN-based networks align better with image classification tasks than MLPs, their representations' predictive power (solid yellow and red lines) are higher than that of MLPs (solid purple lines) on most noise ratios.

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 (He et al., 2016). 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.

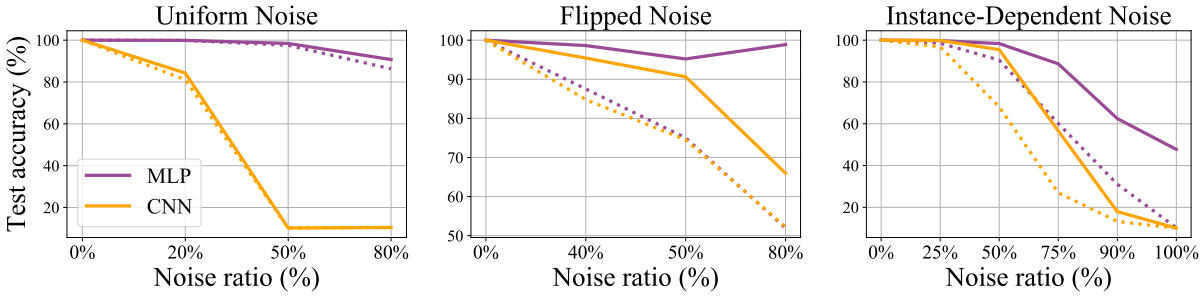


Figure 7. **CIFAR-Easy with uniform, flipped, and instance-dependent label noise.** Each line indicates the raw test accuracy (dotted) and the predictive power in representations (solid) learned by a model trained across various noise ratios. As MLPs align better with the target function than CNN-based 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.

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 (Li et al., 2020)	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 (Li et al., 2020)	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 (Li et al., 2020)	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

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 (LeCun et al., 1995; Kheradpisheh et al., 2016), CNN-based networks are expected to *align better* with the target functions than MLPs on image classification datasets (e.g., CIFAR-10/100).

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 5). 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 1. 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 6) 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 7).

We also observe that *models with similar test performance could have various levels of predictive powers in their learned representations*. For example, in Figure 7, while the test accuracies of MLPs and CNNs are very similar on CIFAR-Easy

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 (Li et al., 2020)	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 (Li et al., 2020)	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

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.

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 D). 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.

C.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 (Li et al., 2020), 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 D.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).

D. Additional Experimental Results

In this section, we include additional experimental results for the predictive power in representations learned (a) under different types off additive label noise (Appendix D.1) and (b) with a robust loss function (Appendix D.2). We further demonstrates that the predictive power in well-aligned networks could even outperform sophisticated methods that also utilize clean labels (Appendix D.3).

D.1. Additive Label Noise on Graph Algorithmic Datasets

We conduct additional experiments on additive label noise drawn from distributions with larger mean and larger variance. We consider four such distributions: Gaussian distributions $\mathcal{N}(10,\, 30)$ and $\mathcal{N}(20,\, 15)$, a long-tailed Gamma distribution with mean equal to 10: $\Gamma(2, \frac{1}{15}) - 20$, and another long-tailed t-distribution with mean equal to 10: $\mathcal{T}(\nu = 1) + 10$. Figure 9 demonstrates that for a GNN well aligned to the target function, its representations are still very predictive even under non-zero mean distributions with larger mean and large variance.

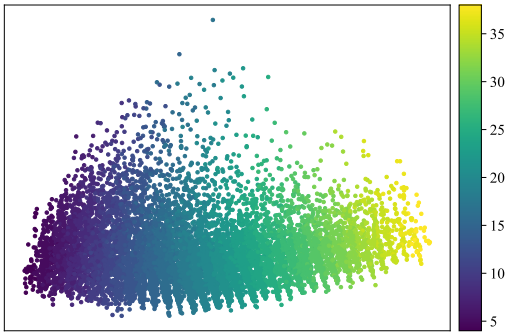


Figure 8. **PCA visualization of hidden representations (colored with true labels)** from a max-sum GNN trained with additive label noise drawn from $\mathcal{N}(10,\ 15)$ at 100% noise ratio. The representations have a clear linear relationship with the true labels.

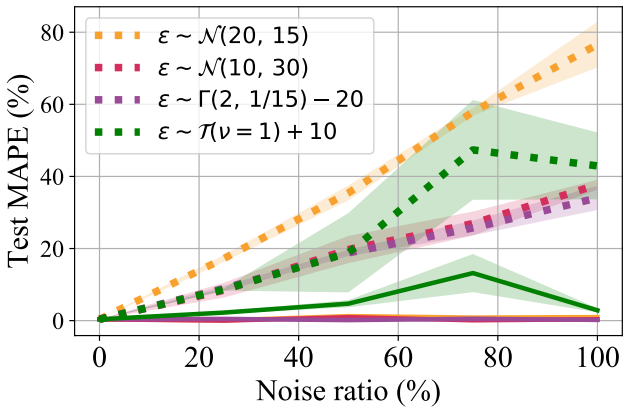


Figure 9. **Additional experiments on simple noise drawn from non-zero mean distributions**. On the maximum degree task, max-sum GNNs have large test errors (dotted lines) under additive label noise drawn from non-zero-mean distributions. Yet, the predictive power in representations (solid lines) greatly reduces the test errors with 10% clean labels.

D.2. Training with a Robust Loss Function

We also train the models with a robust loss function–Mean Absolute Error (MAE), and we observe similar trends in the representations' predictive power as training the models using MSE (Figure 10).

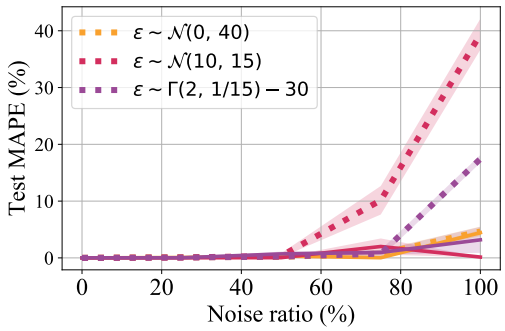
$$\text{loss} = \sum_{i=1}^{n} |y_{\text{true}} - y_{\text{pred}}|. \tag{4}$$

D.3. Comparing with Sophisticated Methods Using Clean Labels

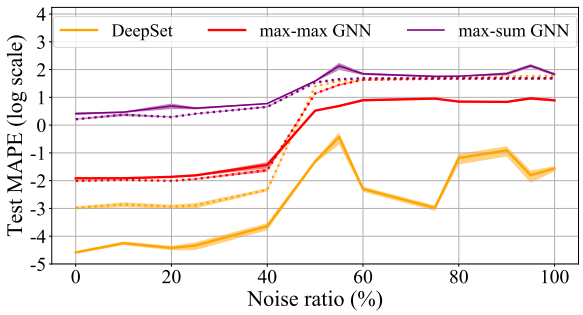
In previous experiments (section C.2), we have shown that the predictive power in well-aligned models could further improve the test performance of SOTA methods on noisy label training. As we use a small set of clean labels to measure the predictive power, we also wonder how the improvements obtained by the predictive power compare with the sophisticated methods that also use clean labels.

D.3.1. Sophisticated Methods Using Clean Labels

In our experiments, we consider the following methods which utilize clean labels: L2R (Ren et al., 2018), MentorNet (Jiang et al., 2018), SELF (Nguyen et al., 2019), GLC (Hendrycks et al., 2018), Meta-Weight-Net (Shu et al., 2019), and IEG (Zhang et al., 2020). Besides, we also compare with training the SOTA method, DivideMix, using clean labels: we mark the set of clean data as labeled data during the semi-supervised learning step in DivideMix. We denote this method as *DivideMix w/ Clean Labels (DwC)*, and we further measure the predictive power in representations learned by DwC.



(a) Test errors of max-sum GNNs on the maximum degree task with additive label noise



(b) Test errors of three different GNNs on the maximum node feature task with instance-dependent label noise

Figure 10. **Predictive power in representations trained with MAE**. For GNNs trained with MAE, the predictive power in representations exhibits similar trends as models trained with MSE. The robust loss function, MAE, is more helpful in learning more predictive representations under smaller noise ratios.

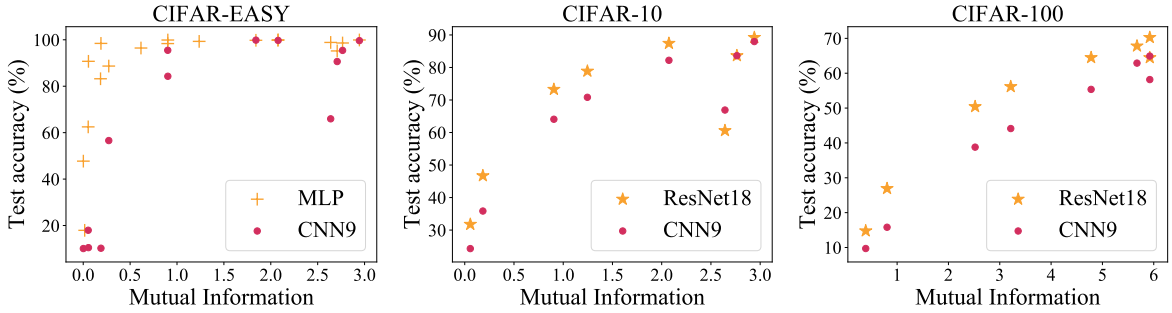


Figure 11. **The predictive in the representations grows as the mutual information between the noisy labels and original clean labels increases** for models well-aligned with the target function. The x-axis of each point in the plots denotes the mutual information between a given noisy dataset and the original clean labels. The corresponding y-axis denotes the representations' predictive power for a model trained on this noisy dataset with standard procedures (i.e., vanilla training).

D.3.2. Datasets

We conduct experiments on CIFAR-10/100 with synthetic noisy labels and on two large-scale datasets with real-world noisy labels: Clothing1M and Webvision.

Clothing1M (Xiao et al., 2015) has real-world noisy labels with an estimated 38.5% noise ratio. The dataset has a small human-verified training data, which we use as clean data. Following recent method (Li et al., 2020), we use 1000 mini-batches in each epoch to train models on Clothing1M.

WebVision (Li et al., 2017) also has real-world noisy labels with an estimated 20% noise ratio. It shares the same 1000 classes as ImageNet (Deng et al., 2009). For a fair comparison, we follow (Jiang et al., 2018) to create a mini WebVision dataset with the top 50 classes from the Google image subset of WebVision. We train all models on mini WebVision dataset and evaluate on both the WebVision and ImageNet validation sets. We select 100 images per class from ImageNet training data as clean data.

D.3.3. Experimental Settings

We use the same architectures and hyperparameters as DivideMix: an 18-layer PreAct Resnet (He et al., 2016) for CIFAR-10/100, a ResNet-50 pre-trained on ImageNet for Clothing1M, and Inception-ResNet-V2 (Szegedy et al., 2016) for WebVision. We use the test accuracy reported in the original papers whenever possible, and the accuracy for L2R (Ren et al., 2018) are from (Zhang et al., 2020). For IEG, we use the reported test accuracy obtained by ResNet-29 rather than WRN28-10, because ResNet-29 has a comparable number of parameters as the PreAct ResNet-18 we use.

As CIFAR-10/100 do not have a validation set, we follow previous works to report the averaged test accuracy over the last 10 epochs: we measure the predictive power in representations for models from these epochs and report the averaged test

Table 3. Test accuracy (%) on **CIFAR-10 with uniform label noise**.

Method	# clean per class	Noise ratio					
		0%	20%	40%	50%	80%	90%
Cross-Entropy	-	95.0 ± 0.1	84.4 ± 0.4	67.9 ± 1.1	58.5 ± 1.5	27.3 ± 0.4	17.2 ± 0.5
IEG (Zhang et al., 2020)	10	94.4	92.9 ± 0.2	92.5 ± 0.5	-	85.6 ± 1.1	-
L2R (Ren et al., 2018)	100	96.1	90.0 ± 0.4	86.9 ± 0.2	-	73.0 ± 0.8	-
MentorNet (Jiang et al., 2018)	500	96.0	92.0	89.0	-	49.0	-
DivideMix (Li et al., 2020)	-	95.0 ± 0.1	95.7	94.4 ± 0.1	94.4	92.9	75.4
DivideMix w/ Clean Labels (DwC)	500	95.0 ± 0.1	95.9 ± 0.1	94.3 ± 0.1	94.8 ± 0.1	92.9 ± 0.2	81.5 ± 0.4
DivideMix's Predictive Power	10	95.0 ± 0.1	**96.0 ± 0.1**	**94.5 ± 0.1**	94.7 ± 0.0	**93.2 ± 0.1**	74.6 ± 0.4
DivideMix's Predictive Power	100	95.0 ± 0.1	**96.1 ± 0.1**	**94.5 ± 0.1**	94.8 ± 0.1	**93.4 ± 0.1**	76.7 ± 0.4
DivideMix's Predictive Power	500	95.0 ± 0.1	**96.1 ± 0.1**	**94.6 ± 0.1**	**94.9 ± 0.1**	**93.6 ± 0.1**	80.4 ± 0.4
DwC's Predictive Power	500	95.0 ± 0.1	**95.9 ± 0.1**	**94.7 ± 0.1**	**95.0 ± 0.1**	**93.5 ± 0.1**	**87.4 ± 0.2**

Table 4. Test accuracy (%) on **CIFAR-10 with flipped label noise**.

Method	# clean per class	Noise ratio		
		20%	40%	80%
Cross-Entropy	-	86.1 ± 0.5	76.9 ± 1.0	54.7 ± 0.7
IEG (Zhang et al., 2020)	10	92.7 ± 0.2	90.2 ± 0.5	78.9 ± 3.5
SELF (Nguyen et al., 2019)	100	92.8	89.1	-
GLC (Hendrycks et al., 2018)	100	89.7 ± 0.3	88.9 ± 0.2	-
Meta-Weight-Net (Shu et al., 2019)	100	90.3 ± 0.6	87.5 ± 0.2	-
DivideMix (Li et al., 2020)	-	94.0 ± 0.3	92.1	56.2 ± 0.1
DivideMix w/ Clean Labels (DwC)	500	94.2 ± 0.2	91.7 ± 0.3	56.9 ± 0.4
DivideMix's Predictive Power	10	93.68 ± 0.34	**92.14 ± 0.50**	**88.71 ± 0.46**
DivideMix's Predictive Power	100	**94.63 ± 0.09**	**93.59 ± 0.12**	**92.68 ± 0.11**
DivideMix's Predictive Power	500	**95.00 ± 0.10**	**94.25 ± 0.11**	**93.87 ± 0.09**
DwC's Predictive Power	500	**94.89 ± 0.11**	**93.53 ± 0.12**	**92.84 ± 0.12**

accuracy. For Clothing1M and Webvision, we use the associated validation set to select the best model and measure the predictive power in its representations.

D.3.4. Results

Tables 3-6 show the results on CIFAR-10 and CIFAR-100 with uniform and flipped label noise, where **boldfaced numbers** denote test accuracies better than all methods we compared with. We see that across different noise ratios on CIFAR-10/100 with flipped label noise, the predictive power in representations remains roughly the same as the test performance of the model trained on clean data for a network well-aligned with the target function, which matches with Lemma 1. For CIFAR-10 with uniform label noise, the predictive power in representations achieves better test performance using only 10 clean labels per class on most noise ratios; for CIFAR-100 with uniform label noise, the predictive power in representations could achieve better test performance using only 50 labels per class.

Table 5. Test accuracy (%) on **CIFAR-100 with uniform label noise**.

Method	# clean per class	Noise ratio					
		0%	20%	40%	50%	80%	90%
Cross-Entropy	-	77.1 ± 0.1	63.2 ± 0.2	49.8 ± 0.3	40.2 ± 0.2	11.5 ± 0.1	3.9 ± 0.1
IEG (Zhang et al., 2020)	10	72.1	69.3 ± 0.5	67.0 ± 0.8	-	60.7 ± 1.0	-
L2R (Ren et al., 2018)	10	81.2	67.1 ± 0.1	61.3 ± 2.0	-	35.1 ± 1.2	-
MentorNet (Jiang et al., 2018)	50	79.0	73.0	68.0	-	35.0	-
DivideMix (Li et al., 2020)	-	77.1 ± 0.1	76.9	74.8 ± 0.2	74.2	59.6	31.0
DivideMix w/ Clean Labels (DwC)	50	77.1 ± 0.1	76.8 ± 0.2	75.0 ± 0.2	74.0 ± 0.2	60.4 ± 0.2	39.8 ± 0.1
DivideMix's Predictive Power	10	77.1 ± 0.1	76.3 ± 0.2	74.0 ± 0.1	73.6 ± 0.2	58.5 ± 0.2	32.6 ± 0.4
DivideMix's Predictive Power	50	77.1 ± 0.1	**77.2 ± 0.2**	**75.1 ± 0.1**	**74.7 ± 0.2**	**61.1 ± 0.1**	37.6 ± 0.3
DwC's Predictive Power	50	77.1 ± 0.1	76.4 ± 0.2	74.6 ± 0.1	73.7 ± 0.2	**61.5 ± 0.1**	**45.1 ± 0.2**

Table 6. Test accuracy (%) on **CIFAR-100 with flipped label noise**.

Method	# clean per class	Noise ratio		
		20%	40%	80%
Cross-Entropy	-	63.6 ± 0.5	45.2 ± 0.3	7.4 ± 0.2
GLC (Hendrycks et al., 2018)	10	63.1 ± 0.5	62.2 ± 0.6	-
Meta-Weight-Net (Shu et al., 2019)	10	64.2 ± 0.3	58.6 ± 0.5	-
DivideMix (Li et al., 2020)	-	77.0 ± 0.2	55.2 ± 0.7	0.2 ± 0.0
DivideMix w/ Clean Labels (DwC)	50	76.9 ± 0.2	55.4 ± 0.8	0.2 ± 0.0
DivideMix's Predictive Power	10	74.31 ± 0.16	$\mathbf{72.09 \pm 0.24}$	$\mathbf{73.75 \pm 0.31}$
DivideMix's Predictive Power	50	76.74 ± 0.18	$\mathbf{74.91 \pm 0.22}$	$\mathbf{76.13 \pm 0.21}$
DwC's Predictive Power	50	76.35 ± 0.18	$\mathbf{74.46 \pm 0.23}$	$\mathbf{75.55 \pm 0.22}$

Table 7. Comparison with state-of-the-art methods in test accuracy (%) on Clothing1M.

Method	# clean	Test Accuracy
Cross-Entropy	-	69.21
DivideMix (Li et al., 2020)	-	74.76
IEG (Zhang et al., 2020)	50k	77.21
CleanNet (Lee et al., 2018)	50k	79.9
F-correction (Patrini et al., 2017)	50k	80.38
Self-learning (Han et al., 2019)	50k	**81.16**
DivideMix+Ours	50k	**80.47**

Moreover, we observe that adding clean data to the labeled set in DivideMix (DwC) may barely improve the model's test performance when the noise ratio is small and under flipped label noise. At 90% uniform label noise, DwC can greatly improve the model's test performance, and the predictive power in representations can achieve a even higher test accuracy with the same set of clean data used to train DwC.

On Clothing1M, we compare the predictive power in representations learned by DivideMix with existing methods that use the small set of human-verified data: CleanNet (Lee et al., 2018), F-correction (Patrini et al., 2017) and Self-learning (Han et al., 2019). As these methods also use the clean subset to fine-tune the whole model, we follow similar procedures to fine-tune the model (trained by DivideMix) for 10 epochs and then select the best model based on the validation accuracy to measure the predictive power in its representations. The predictive power in representations could further improve the test accuracy of DivideMix by around 6% and outperform IEG, CleanNet, and F-correction (Table 7). The improved test accuracy is also competitive to (Han et al., 2019), which uses a much more complicated learning framework.

On Webvision, the predictive power also improves the model's test performance (Table 8). The improvement is less significant than on Clothing1M as the estimated noise ratio on Webvision (20%) is smaller than Clothing1M (38.5%).

E. Experimental Details

E.1. Measuring the Predictive Power

We use linear regression to train the linear model when measuring the predictive power in representations. For representations from all models except MLPs, we use ordinary least squares linear regression (OLS). When the learned representations are from MLPs, we e use ridge regression with penalty = 1 since we find the linear models trained by OLS may easily overfit to the small set of clean labels.

E.2. Experimental Details on GNNs

Common settings. In the generated datasets, each graph $\mathcal{G}$ is sampled from Erdős-Rényi random graphs with an edge probability uniformly chosen from $\{0.1, 0.2, \cdots, 0.9\}$. This sampling procedure generates diverse graph structures. The training and validation sets contain 10,000 and 2,000 graphs respectively, and the number of nodes in each graph is randomly picked from $\{20, 21, \cdots, 40\}$. The test set contains 10,000 graphs, and the number of nodes in each graph is randomly

Table 8. Comparison with state-of-the-art methods trained on (mini) WebVision dataset. Numbers denote top-1 (top-5) accuracy (%) on the WebVision and the ImageNet validation sets.

Method	WebVision		ILSVRC12	
	top1	top5	top1	top5
MentorNet (Jiang et al., 2018)	63.00	81.40	57.80	79.92
IEG (Zhang et al., 2020)	-	-	**80.0**	**94.9**
DivideMix (Li et al., 2020)	77.32	91.64	75.20	90.84
DivideMix+Ours	**77.70 ± 0.23**	90.68	75.99 ± 0.09	91.30

picked from $\{50, 51, \cdots, 70\}$.

E.2.1. Additive Label Noise

Dataset Details. In each graph, the node feature $\boldsymbol{x}_u$ is a scalar randomly drawn from $\{1, 2, \cdots, 100\}$ for all $u \in \mathcal{G}$.

Model and hyperparameter settings. We consider a 2-layer GNN with max-aggregation and sum-readout (max-sum GNN):

$$\boldsymbol{h}_G = \text{MLP}^{(2)}\Big(\max_{u\in\mathcal{G}} \sum_{v\in\mathcal{N}(u)} \text{MLP}^{(1)}\Big(\boldsymbol{h}_u, \boldsymbol{h}_v\Big)\Big), \boldsymbol{h}_u = \sum_{v\in\mathcal{N}(u)} \text{MLP}^{(0)}\Big(\boldsymbol{x}_u, \boldsymbol{x}_v\Big).$$

The width of all MLP modules are set to 128. The number of layers are set to 3 for $\text{MLP}^{(0)}$ and $\text{MLP}^{(1)}$. The number of layers are set to 1 for $\text{MLP}^{(2)}$. We train the max-sum GNNs with loss function MSE or MAE for 200 epochs. We use the Adam optimizer with default parameters, zero weight decay, and initial learning rate set to 0.001. The batch size is set to 64. We early-stop based on a noisy validation set.

E.2.2. Instance-Dependent Label Noise.

Dataset Details. Since the task is to predict the maximum node feature and we use the maximum degree as the noisy label, the correlation between true labels and noisy labels are very high on large and dense graphs if the node features are uniformly sampled from $\{1, 2, \cdots, 100\}$. To avoid this, we use a two-step method to sample the node features. For each graph $\mathcal{G}$, we first sample a constant upper-bound $M_{\mathcal{G}}$ uniformly from $\{20, 21, \cdots, 100\}$. For each node $u \in \mathcal{G}$, the node feature $\boldsymbol{x}_u$ is then drawn from $\{1, 2, \cdots, M_{\mathcal{G}}\}$.

Model and hyperparameter settings. We consider a 2-layer GNN with max-aggregation and sum-readout (max-sum GNN), a 2-layer GNN with max-aggregation and max-readout (max-max GNN), and a special GNN (DeepSet) that does not use edge information:

$$h_G = \text{MLP}^{(1)}\Big(\max_{u\in\mathcal{G}} \text{MLP}^{(0)}\Big(\boldsymbol{x}_u\Big)\Big).$$

The width of all MLP modules are set to 128. The number of layers is set to 3 for $\text{MLP}^{(0)}, \text{MLP}^{(1)}$ in max-max and max-sum GNNs and for $\text{MLP}^{(0)}$ in DeepSet. The number of layers is set to 1 for $\text{MLP}^{(2)}$ in max-max and max-sum GNNs and for $\text{MLP}^{(1)}$ in DeepSet. We train these GNNs with MSE or MAE as the loss function for 600 epochs. We use the Adam optimizer with zero weight decay. We set the initial learning rate to 0.005 for DeepSet and 0.001 for max-max GNNs and max-sum GNNs. The models are selected from the last epoch so that they can overfit the noisy labels more.

E.3. Experimental Details on Vision Datasets

Neural Network Architectures. Table 9 describes the 9-layer CNN (Miyato et al., 2018) used on CIFAR-Easy and CIFAR-10/100, which contains 9 convolutional layers and 19 trainable layers in total. Table 10 describes the 4-layer MLP used on CIFAR-Easy and CIFAR-10/100, which has 4 linear layers and ReLU as the activation function.

Vanilla Training. For models trained with standard procedures, we use SGD with a momentum of 0.9, a weight decay of 0.0005, and a batch size of 128. For ResNets and CNNs, the initial learning rate is set to 0.1 on CIFAR-10/100 and 0.01 on CIFAR-Easy. For MLPs, the initial learning rate is set to 0.01 on CIFAR-10/100 and 0.001 on CIFAR-Easy. The

Table 9. 9-layer CNN on CIFAR-Easy and CIFAR-10/100.

Input	32×32 Color Image
Block 1	Conv(3×3, 128)-BN-LReLU Conv(3×3, 128)-BN-LReLU Conv(3×3, 128)-BN-LReLU MaxPool(2×2, stride = 2) Dropout(p = 0.25)
Block 2	Conv(3×3, 256)-BN-LReLU Conv(3×3, 256)-BN-LReLU Conv(3×3, 256)-BN-LReLU MaxPool(2×2, stride = 2) Dropout(p = 0.25)
Block 3	Conv(3×3, 512)-BN-LReLU Conv(3×3, 256)-BN-LReLU Conv(3×3, 128)-BN-LReLU GlobalAvgPool(128)
Score	Linear(128, 10 or 100)

Table 10. 4-layer FC on CIFAR-Easy and CIFAR-10/100.

Input	32×32 Color Image
Block 1	Linear(32×32×3, 512)-ReLU Linear(512, 512)-ReLU Linear(512, 512-ReLU
Score	Linear(512, 10 or 100)

initial learning rate is multiplied by 0.99 per epoch on CIFAR-10/100, and it is decayed by 10 after 150 and 225 epochs on CIFAR-Easy.

Train Models with SOTA Methods. We use the same set of hyperparameter settings from DivideMix (Li et al., 2020) to obtain corresponding trained models and measure the predictive power in representations from these models.

On CIFAR-10/100 with flipped noise, we only use the small set of clean labels to train the linear model in our method, and the clean subset is randomly selected from the training data. On CIFAR-10/100 with uniform noise, the clean labels we use are from examples with highest model uncertainty (Lewis & Gale, 1994). Besides the clean set, we also use randomly-sampled training examples labeled with the model's original predictions to train the linear model. We use 5,000 such samples under 20%, 40%, 50%, and 80% noise ratios, and we use 500 such samples under 90% noise ratio.

F. Theoretical Results

We first provide a formal version of Theorem 1 based on (Xu et al., 2020a), which connects the predictive power with alignment *when the network is trained on clean data*.

Theorem 2. *(Better alignment implies better predictive power on clean training data; (Xu et al., 2020a)). Fix ϵ and δ. Given a target function $f : \mathcal{X} \to \mathcal{Y}$ that can be decomposed into functions $f_1, ..., f_n$ and a network $\mathcal{N}$, where $\mathcal{N}_1, ..., \mathcal{N}_n$ are $\mathcal{N}$'s modules in sequential order. Suppose the training dataset $\mathcal{S} := \{\boldsymbol{x}_j, y_j\}_{j=1}^M$ contains i.i.d. samples drawn from a distribution with clean labels $y_j := f(x_j)$. Then under the following assumptions, Alignment$(\mathcal{N}, f, \epsilon, \delta) \leq M$ if and only if there exists a learning algorithm A such that the network's last module $\mathcal{N}_n$'s representations learned by A on the training data $\mathcal{S}$ have predictive power $P_n(f, \mathcal{N}, \mathcal{S}) \leq \epsilon$ with probability $1 - \delta$.*

Assumptions:

(a) *We train each module $\mathcal{N}_i$'s sequentially: for each $\mathcal{N}_i$, the input samples are $\{h^{(i-1)}(\boldsymbol{x}_j), f_i(h^{(i-1)}(\boldsymbol{x}_j))\}_{j=1}^M$ with $h^{(0)}(\boldsymbol{x}) = \boldsymbol{x}$. Notice that each input $h^{(i-1)}(\boldsymbol{x}_j)$ is the output from the previous modules, but its label is generated by the function f_i on $h^{(i-1)}(\boldsymbol{x}_j)$.*

(b) *For the clean training set $\mathcal{S}$, let $\mathcal{S}' := \{\hat{\boldsymbol{x}}_j, y_j\}_{j=1}^M$ denote the perturbed training data (with the same labels). Let $f_{\mathcal{N},A}$ and $f'_{\mathcal{N},A}$ denote the functions obtained by the learning algorithm A operating on $\mathcal{S}$ and $\mathcal{S}'$ respectively. Then for any $\boldsymbol{x} \in \mathcal{X}$, $\|f_{\mathcal{N},A}(\boldsymbol{x}) - f'_{\mathcal{N},A}(\boldsymbol{x})\| \leq L_0 \cdot \max_{\boldsymbol{x}_j \in \mathcal{S}} \|\boldsymbol{x}_j - \hat{\boldsymbol{x}}_j\|$, for some L_0.*

(c) *For each module $\mathcal{N}_i$, let $\hat{f}_i$ denotes its corresponding function learned by the algorithm A. Then for any $\boldsymbol{x}, \hat{\boldsymbol{x}} \in \mathcal{X}$, $\|\hat{f}_j(\boldsymbol{x}) - \hat{f}_j(\hat{\boldsymbol{x}})\| \leq L_1 \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|$, for some L_1.*

The experiments in this work have empirically shown that Theorem 2 may also hold when the training data has noisy labels. To also provide some theoretical support, we prove Theorem 2 for a simplified noisy setting where the target function and

noise function share a common feature space, but have different prediction rules. For example, the target function and noise function share the same feature space under flipped label noise (in classification setting). Yet, they have different mappings from the learned features to labels associated.

Lemma 1. *(Better alignment implies better predictive power on noisy training data). Fix ϵ and δ. Let $\{\boldsymbol{x}_j\}_{j=1}^{M}$ be i.i.d. samples drawn from a distribution. Given a target function $f : \mathcal{X} \to \mathcal{Y}$ and a noise function $g : \mathcal{X} \to \mathcal{Y}$, let $y := f(\boldsymbol{x})$ denote the true label for an input $\boldsymbol{x}$, and $\hat{y} := g(\boldsymbol{x})$ denote the noisy label of $\boldsymbol{x}$. Let $\hat{\mathcal{S}} := \{(\boldsymbol{x}_j, y_j)\}_{j=1}^{N} \bigcup \{(\boldsymbol{x}_j, \hat{y}_j)\}_{j=N+1}^{M}$ denote a noisy training set with $M - N$ noisy samples for some $N \in \{1, 2, \cdots, M\}$. Given a network $\mathcal{N}$ with modules $\mathcal{N}_i$, suppose the alignment between $\mathcal{N}$ and f is less than M (i.e., Alignment$(\mathcal{N}, f, \epsilon, \delta) \leq M$). Then under the same assumptions of Theorem 2 and the additional assumptions below, there exists a learning algorithm A and a module $\mathcal{N}_i$ such that when training the network $\mathcal{N}$ on the noisy data $\hat{\mathcal{S}}$ with algorithm A, the representations from its i-th module have predictive power $P_i(f, \mathcal{N}, \mathcal{C}) \leq \epsilon$ with probability $1 - \delta$, where $\mathcal{C}$ is a small set of clean data and of size greater than the number of dimensions of module $\mathcal{N}_i$'s output.*

Additional assumptions (a simplified noisy setting):

(a) *There exists a function h on the domain $\mathcal{X}$ such that the target function $f : \mathcal{X} \to \mathcal{Y}$ and the noise function $g : \mathcal{X} \to \mathcal{Y}$ can be decomposed as: $f(\boldsymbol{x}) = f_r(h(\boldsymbol{x}))$ with f_r being a linear function and $g(\boldsymbol{x}) = g_r(h(\boldsymbol{x}))$ for some function g_r.*
(b) *f_r is a linear map from a high-dimensional space to a low-dimensional space.*
(c) *The loss function used in measuring the predictive power is mean squared error (denoted as $\|\cdot\|$) .*

Remark. Lemma 1 suggests that the representations' predictive power for models well aligned with the target function should remain roughly similar across different noise ratios under flipped label noise. We observe phenomenons similar to this in Figures 6-7, and in Tables 4 and 6. Some discrepancy between the experimental and theoretical results could exist under vanilla training as Lemma 1 assumes sequential training, which is quite different from standard training procedures.

Proof of Lemma 1. Based on the definition of alignment, since *Alignment*$(\mathcal{N}, f, \epsilon, \delta) \leq M$ and $f(\boldsymbol{x}) = f_r(h(\boldsymbol{x}))$, we can find a sub-structure (denoted as $\mathcal{N}_{sub}$) in the network $\mathcal{N}$ with sequential modules $\{\mathcal{N}_1, \cdots, \mathcal{N}_i\}$ such that the sample complexity for $\mathcal{N}_{sub}$ to learn h is no larger than M. According to Theorem 2, applying sequential learning to train $\mathcal{N}_{sub}$ with labels $h(\boldsymbol{x})$, the representations of $\mathcal{N}_{sub}$ will have predictive power $P_i(h, \mathcal{N}_{sub}, \mathcal{C}) \leq \epsilon$ with probability $1 - \delta$.

Since for each input $\boldsymbol{x}$ in the noisy training data $\hat{\mathcal{S}}$, its label can be written as $f_r(h(\boldsymbol{x}))$ (if it is clean) or $g_r(h(\boldsymbol{x}))$ (if it is noisy), when the network $\mathcal{N}$ is trained on $\hat{\mathcal{S}}$ using sequential learning, its sub-structure $\mathcal{N}_{sub}$ can still learn h efficiently (i.e., $\mathcal{M}_A(h, \mathcal{N}_{sub}, \epsilon, \delta) \leq M$ for some learning algorithm A), and thus have predictive power in representations $P_i(h, \mathcal{N}_{sub}, \mathcal{C}) \leq \epsilon$ with probability $1 - \delta$.

Since f_r is a linear map from a high-dimensional space to a low-dimensional space, and the clean data $\mathcal{C}$ has greater number of samples than the input dimension of f_r, the linear regression has a closed form solution (as the problem is over-complete) and the learned linear model L can generalize f_r. Therefore, for the situations where $P_i(h, \mathcal{N}_{sub}, \mathcal{C}) \leq \epsilon$, $P_i(f, \mathcal{N}_{sub}, \mathcal{C}) \leq \epsilon$ also holds. Notice that $P_i(f, \mathcal{N}_{sub}, \mathcal{C}) = P_i(f, \mathcal{N}, \mathcal{C})$ as $\mathcal{N}_i$ is also the i-th module in $\mathcal{N}$. Hence, we have shown that $P_i(f, \mathcal{N}, \mathcal{C}) \leq \epsilon$ with probability $1 - \delta$.