A Nonlinear, Noise-aware, Quasi-clustering Approach to Learning Deep CNNs from Noisy Labels

Ishan Jindal
Wayne State University
Detroit MI
ishan.jindal@wayne.edu

Matthew Nokleby
Wayne State University
Detroit MI
matthew.nokleby@wayne.edu

Daniel Pressel
Interactions Digital Roots
Ann Arbor MI
dpressel@interactions.com

Xuewen Chen
Wayne State University
Detroit MI
xwchen@wayne.edu

Abstract

The success of deep convolutional networks on image classification and recognition tasks depends on the availability of large, labeled datasets, but it can be difficult to obtain a large number of accurate labels. To deal with this problem, we present Nonlinear, Noise-aware, Quasi-clustering *(NNAQC), a method for learning deep convolutional networks from datasets corrupted by unknown label noise. We append a nonlinear noise model to a standard convolutional network, which is learned in tandem with the parameters of the network. Further, we train the network using a loss function that encourages the clustering of training images. We argue that the non-linear noise model, while not rigorous as a probabilistic model, results in a more effective denoising operator during backpropagation. We evaluate the performance of NNAQC on artificially injected label noise to MNIST, CIFAR-10, CIFAR-100 and ImageNet datasets and on a large-scale Clothing1M dataset with inherent label noise. On all these datasets, NNAQC provides significantly improved classification performance over the state of the art and is robust to the amount of label noise and the training samples.*

1. Introduction

The last decade has seen dramatic advances in image classification, image captioning, object recognition, and more, owing mostly to deep convolutional neural networks (CNNs) trained on large, labeled datasets [13, 18, 29]. Researchers often benchmark the performance of these algorithms on standard, curated datasets such as MNIST, CIFAR, ImageNet, or MSCOCO [15, 12, 6, 18]. However, use of curated data sets elides a crucial point: in practical datasets, labels are not always reliable. "Crowdsourced" labels obtained from social media or other non-expert sources are subject to error, and in subjective tasks even humans or experts may disagree on the correct label. (For a taxonomy of types and sources of label noise, see [7] and the references therein.) As deep learning systems become more complex and are trained on even more massive datasets, it becomes increasingly difficult to obtain clean labels. In this scenario, an approach to learning that accounts for noisy labels is needed.

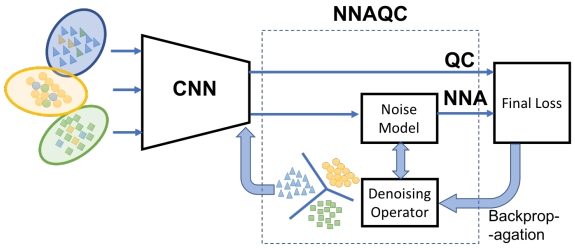

Figure 1: NNAQC architecture for learning deep CNNs from noisy labels. The learned Noise model act as a denosing operator while backpropagation.

In this paper, we present a two-pronged approach, as shown in Fig. 1, to learning CNNs from training sets corrupted by label noise having unknown statistics. The first prong is to augment the CNN architecture with a nonlinear model for the label noise, which is learned during training. A challenge with this approach is that the model is underdetermined: when CNN outputs disagree with the labels provided, it is not clear whether the CNN should update its weights to match the noisy labels or update the noise model to account for the possibility of incorrect labels. The non-

linear model turns out to be particularly effective in handling these situations. We show that the proposed model is actually non-rigorous as a transition probability between clean and noisy labels; however, it results in a "denoising" operator that better handles errors when training the CNN via backpropagation. Because the noise model is not used at test time, learning an accurate and rigorous noise model is less important than the impact of the noise model on CNN training. The second prong is to augment the standard cross-entropy loss with a term that encourages the CNN to cluster images in feature space. This allows the network to learn from the natural clustering of the data, even when labels are unreliable. We term the combined approach *Nonlinear, Noise-aware, Quasi-clustered* learning (NNAQC).

We demonstrate the performance of NNAQC on the MNIST, CIFAR-10, CIFAR-100 and ImageNet datasets corrupted by label noise. On these datasets, NNAQC exhibits state-of-the-art performance in terms of classification accuracy over a clean test set. The results show that NNAQC is scalable to a large number of image categories. The results are robust to label noise, achieving near-optimum performance when there is little noise, and maintaining classification accuracy as the label noise increases. We emphasize that this robustness does not require knowledge of the label noise statistics or tuning of hyperparameters. The performance of NNAQC degrades gracefully as the training size decreases, suggesting that it sufficiently regularizes the learning of the combined noise and classification model. Indeed, in some cases the performance is *better* when using fewer training samples, which suggests that the sample complexity of the model is occasionally too high for the given dataset, in which case regularization in the form of early stopping improves performance. Finally, we evaluate NNAQC on Clothing1M [33] dataset, consisting of 1M images with noisy labels.

The rest of the paper is organized as follows. First, we review the existing literature on this subject in Section 2. Then, we describe the details of the proposed approach NNAQC and its different components in Section 3. Finally, in Section 4 we present experimental results on variety of datasets and conclude our work with future directions.

2. Relationship to Prior Work

This problem is closely related to semi-supervised and weakly-supervised learning, for which there is an extensive body of work. We refer the reader to [34] for survey.

Previous work addresses the question of learnability when labels are binary and label noise is i.i.d. and class-independent [1], and provides sample complexity bounds in terms of the VC dimension for the 0-1 loss. More recently, [23] provides sample complexity bounds for more general loss functions, in terms of the Rademacher complexity, for *class-conditional* label noise having known statistics. The upshot of these works is that if labels are flipped with probability η, the sample complexity increases roughly by a factor of $1/(1-2\eta)^2$. Equivalently, the generalization error scales roughly as $1/(\sqrt{n}(1-2\eta))$ instead of the usual $\sqrt{1/n}$. Other possible solution to this problem includes estimation of the noise rate. A class conditional estimator for estimating the noise rate is proposed in [19].

Earlier works consider label noise for general learning algorithms. For example, [14] presents a method for learning a kernel-based classifier from noisy labels with unknown statistics. Using an EM-style algorithm, their approach learns jointly a generative noise model and a classifier. Similarly, [25] employs an EM-style algorithm to estimate the reliability of labels. Other techniques detect and discard samples with anomalous labels [5] or relabel erroneous samples [4]. In a similar vein to [23], a recent work shows that careful choice of the loss function leads to learning that is provably robust to label noise [22].

Recently, authors have begun designing CNNs to deal with label noise for image classification [20, 32] and text classification [11, 24]. In some of the approaches [27, 10], a standard CNN is augmented with a generative noise model that must be learned in tandem with the CNN parameters. A joint optimization framework presented in [28] simultaneously learns the parameters and estimates the true labels. As mentioned above, the augmented model is underdetermined and must be regularized, else the network may choose the identity as the noise transition matrix. Each of these works imposes a different regularization term to encourage a non-trivial noise model: [27] imposes a cost on the trace of the label noise transition probability matrix, whereas [10] uses Dropout regularization. In [17], an unified distillation framework is proposed to learn CNN from the noisy labels. This framework uses label relations in knowledge graphs and a small clean dataset to learn a classifier from noisy labels. [21] proposes to train in parallel two neural networks, which weights are updated only when label predictions disagree.

A recent stream of work exploits an additional model (again, a neural network) to denoisify /take into account the confidence of prediction on noisy examples [33, 31]. Similar to [23], [24] estimates a noise model in a theoretical motivated manner during a pre-training phase of the network, and then correct the loss function. It estimates the transition matrix heuristically but with the large number of classes this estimation is not easy to obtain. On CIFAR-10, [27] provides competitive performance as long as the label noise is not too strong. Rather than learning a noise model directly, [26] employs a bootstrapping-style approach in which the loss function encourages consistent predictions for similar input images. In a similar vein, [3] identifies and discards outliers in order to fine-tune a pretrained CNN.

The proposed NNAQC incorporates the spirit of [27,

10]—in that it learns an explicit noise model—and clusters the data via an EM-style approach. The intuition is to identify mislabeled images by their inconsistency with similar images, which combats label noise and emulates unsupervised learning. The upshot of this work is that a label noise model is beneficial, especially when it is regularized by an unsupervised component in the loss function. However, learning a correct noise model is neither necessary nor sufficient for state-of-the-art performance. Indeed, NNAQC uniformly outperforms a genie-aided CNN, similar to [10]. NNAQC denoise the gradient of the loss by a denoising operator before being fed into the gradient of the base model parameters. Our approach produces a very diffuse denoising operator and thus prevents the base model from learning the noisy label directly.

3. Proposed Approach

We consider the supervised learning of a classifier of d-dimensional images that belong to one of L image classes. Let the (noise-free) training set be denoted by $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$, where $x_i \in \mathbb{R}^d$ is the ith image and $y_i \in \{1, \ldots, L\}$ is its label, and where implicitly there is an unknown joint distribution $p(x, y)$ on the image/label pairs. Ideally, one would train a classifier on the training set $\mathcal{D}$, but we suppose that instead of access to the noise-free training set $\mathcal{D}$, we obtain a training set with unreliable labels. Let this *noisy* training set be denoted by $\mathcal{D}' = \{(x_1, y_1'), (x_2, y_2'), \cdots, (x_n, y_n')\}$, where y_i' is a potentially erroneous label for x_i. We suppose *class-conditional* label noise, where the noisy label y_i' depends only on the true label y_i, but not on the image x_i or any other labels y_j or y_j'. Under this model, the label noise is characterized by the conditional distribution $p(y'|y)$, which we describe via the $L \times L$ column-stochastic matrix ϕ, with $\phi_{ij} = p(y' = j|y = i)$. We use a noise model parameterized by the overall probability of a label error, denoted by $0 \leq p \leq 1$:

$$\phi = (1 - p)\mathbf{I} + p\Delta, \tag{1}$$

where $\mathbf{I}$ is the identity matrix, and Δ is a matrix with zeros along the diagonal and remaining entries of each column are drawn uniformly and independently from the $L - 1$-dimensional unit simplex. That is, the label error probability for each class is p, while the probability distribution *within* the erroneous classes is drawn uniformly at random.

Our objective is to train a CNN, using the noisy set $\mathcal{D}'$, that makes accurate predictions of the true label y given an input image x. It is straightforward to train a CNN that predicts the *noisy* labels.The conditional distribution for the noisy label of the image x can be written as:

$$p(y' = \hat{y}'|x) = \sum_i p(y' = \hat{y}'|y = \hat{y}_i)p(y = \hat{y}_i|x). \tag{2}$$

One can learn the classifier associated with $p(y' = \hat{y}'|x)$ via standard training on the noisy set $\mathcal{D}'$. To predict the clean labels, i.e. to learn the conditional distribution $p(y = \hat{y}_i|x)$ requires more effort, as we cannot extract the "clean" classifier from the noisy classifier when the label noise distribution is unknown.

3.1. NNAQC

The architecture of our proposed framework is shown in Fig. 2. We take a standard deep CNN—which we call the *base model*—and augment it with a model that accounts for the label noise. The base and noise models are trained jointly using $\mathcal{D}'$ via stochastic gradient descent. The noise model is used only during training, in which it effectively "denoises" the gradients associated with the noisy labels during backpropagation in order to improve the learning of the base model. Because there is no need to predict the noisy labels of test images, we disconnect the noise model at test time and classify using the base model alone.

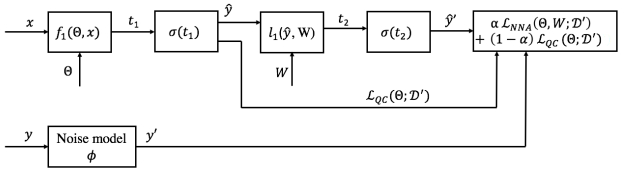

Figure 2: The nonlinear, noise-aware, quasi-clustering (NNAQC) framework.

We stack an additional one fully-connected processing layer [1] on top of base CNN model. We lump the base model parameters—processing layer weights and biases, etc.—into a parameter vector Θ. The high-level features that the base model outputs, which we denote via $t_1(x; \Theta) \in \mathbb{R}^L$, are put through the usual softmax function: $\sigma(z)_i = \frac{\exp(z_i)}{\sum_{j=1}^{L} \exp(z_i)}$ to produce the conditional distribution of the clean label; i.e. $p(y|x; \Theta) = \sigma(t_1(x; \Theta))$, from which we can predict the clean label of an image x. A distinct feature of the proposed approach is that we use a *nonlinear* transformation between the estimate of the clean labels and the estimate of the noisy labels. In an abuse of notation, let $\hat{y} = \sigma(t_1)$ denote the probabilities $p(y|x; \Theta)$. To obtain the probabilities of the noisy labels, denoted $\hat{y}'$, we perform a softmax regression on $\hat{y}$:

$$p(y'|x; \Theta, W) := \sigma(W\sigma(t_1(x; \Theta))), \tag{3}$$

where $W \in \mathbb{R}^{L \times L}$ is a square matrix that governs the transition probabilities. We emphasize a subtle point: This formulation is not rigorous probabilistically. Equation (3) does

[1] We emphasize that the NNAQC framework can be applied to any CNN architecture.

not correctly compute marginal probability of the noisy labels according to (2). To be consistent with the law of total probability, we *should* calculate conditional distribution as

$$p(y' = i|y = j; W) = [\sigma(We_j)]_i, \tag{4}$$

where e_j is the jth elementary vector. From this conditional distribution, the distribution on y' should be

$$p(y'|x; \Theta, W) = \sigma(W)\sigma(t_1(x; \Theta)), \tag{5}$$

where $\sigma(W)$ is the softmax function applied to each column of W. This is equivalent to the architecture used in [27], where a simple linear layer with column-stochastic weight matrix is learned, ideally to match the matrix ϕ that governs the label noise. By taking a nonlinear transformation of $\hat{y}$, instead of a linear transformation associated with transition probabilities, we violate the laws of probability in computing $\hat{y}'$. Nevertheless, empirically we see that the resulting classifier has excellent performance, and in the next part we give a justification for this approach.

Now, the challenge is to learn jointly the CNN parameters Θ and the nonlinear noise model parameters W. One approach is to minimize the standard cross-entropy loss of the end-to-end model, which we call the *nonlinear noise-aware* loss $\mathcal{L}_{\text{NNA}}$:

$$\begin{aligned}\mathcal{L}_{\text{NNA}}(\Theta, W; \mathcal{D}') &= -\frac{1}{n}\sum_{i=1}^{n} \log p(y' = \hat{y}'_i|x_i; \Theta, W) \\ &= -\frac{1}{n}\sum_{i=1}^{n} \log[\sigma(W\sigma(t_1(x_i; \Theta)))]_{\hat{y}_i}.\end{aligned}$$

Empirically we see that this loss function leads to quite good predictions of the true labels. However, $\mathcal{L}_{\text{NNA}}$ does not directly encourage the model to predict correctly the true label $\hat{y}$ as the true label; instead, the prediction of $\hat{y}$ is judged only indirectly via the noisy label predictions $\hat{y}'$. Indeed, this approach treats $\hat{y}$ as an additional hidden layer that acts as an information bottleneck.

To encourage good predictions of $\hat{y}$, we need to feed $\hat{y}$ into the loss function directly. To do so, we introduce an additional term that encourages a "quasi-clustering" of the training images. Images that are close in feature space usually will have the same label, a fact that we can exploit when dealing with noisy labels. We penalize the cross-entropy between a linear combination of the predicted labels and the noisy labels and the predicted labels themselves, i.e.

$$\begin{aligned}\mathcal{L}_{\text{QC}}(\Theta; \mathcal{D}') &= -\frac{1}{n}\sum_{i=1}^{n} (\beta p(\hat{y}_i|x_i; \Theta) + (1-\beta)y'_i) \\ &\quad \times \log p(\hat{y}_i|x_i; \Theta), \\ &= -\frac{1}{n}\sum_{i=1}^{n} (\beta[\sigma(t_1(x; \Theta))]_{\hat{y}_i} + (1-\beta)y'_i) \\ &\quad \times \log[\sigma(t_1(x; \Theta))]_{\hat{y}_i}.\end{aligned}$$

This type of loss function has been used widely in the literature, such as in [8, 26, 2], and it has the effect of clustering the data. For a large value of β, minimizing this loss function encourages $\hat{y}$ toward a low-entropy vector, i.e. one with most of its mass on a single point. In order to make such confident predictions, the CNN needs to map similar output features to similar classes, which is equivalent to clustering. Finally, we form the *nonlinear, noise-aware, quasi-clustering* loss, denoted $\mathcal{L}$, by taking a convex combination of the two losses:

$$\mathcal{L}(\Theta, W; \mathcal{D}') = \alpha\mathcal{L}_{\text{NNA}}(\Theta, W; \mathcal{D}') + (1-\alpha)\mathcal{L}_{\text{QC}}(\Theta; \mathcal{D}'), \tag{6}$$

and we minimize the NNAQC loss via standard back-propagation over the noisy training set $\mathcal{D}'$. We obtain the values of α and β via cross-validation.

3.2. Justifying the Nonlinear Noise Model

In this section, we study the effect of the proposed nonlinear noise model. At first instance, it seems that we are violating the basic laws of probability by adding a nonlinear softmax layer at the output as described above. We emphasize, however, that the role of the noise model is *not* to make accurate predictions of the noisy labels, but to encourage the learning of a CNN that makes accurate predictions of the clean labels instead of noisy ones. Therefore, the ultimate test of a noise model is the extent to which it improves training. To that end, the NNAQC architecture is designed not to learn an explicit noise model, but to learn a "denoising" operator that effectively filters the gradients associated with the noisy labels. To see the benefits of this approach, we examine the back-propagation gradient steps for the base model parameters for the NNAQC architecture and for a CNN augmented with a standard linear noise model.

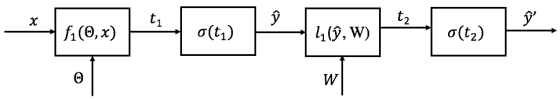

Figure 3: Augmented linear layer with Softmax

In Fig. 3, we zoom in on the NNAQC architecture. For an input sample x, we lump all the initial convolutional, ReLu and pooling layers into one function $f_1(\Theta, x)$ with parameters Θ, and we obtain the normalized prediction of true labels $\hat{y}$ via the first softmax layer $\sigma(t_1)$. We pass the clean label predictions through the matrix W and take the softmax function to obtain the noise label distribution $\hat{y}'$. To observe the effect of the prediction $\hat{y}$ and the noise model parameters W on the learning process, we write down the

gradient of the loss function $\mathcal{L}$ with respect to Θ:

$$\frac{\partial \mathcal{L}}{\partial \Theta} = \frac{\partial \mathcal{L}}{\partial \hat{y}'} \frac{\partial \sigma(t_2)}{\partial t_2} \frac{\partial l_1(\hat{y}, W)}{\partial \hat{y}} \frac{\partial \sigma(t_1)}{\partial t_1} \frac{\partial f_1(\Theta, x)}{\partial \Theta} \quad (7)$$

$$= \frac{\partial \mathcal{L}}{\partial \hat{y}'} \left(\frac{\partial \sigma(t_2)}{\partial t_2} W \frac{\partial \sigma(t_1)}{\partial t_1} \right) \frac{\partial f_1(\Theta, x)}{\partial \Theta}. \quad (8)$$

For comparison, in Fig. 4 we consider a linear noise model as described in (4)-(5), where the matrix W determines the noise model via the stochastic matrix $\sigma(W)$. We write the gradient steps similar to the previous case as:

$$\frac{\partial \mathcal{L}}{\partial \Theta} = \frac{\partial \mathcal{L}}{\partial \hat{y}'} \frac{\partial l_1(\hat{y}, \sigma(W))}{\partial \hat{y}} \frac{\partial \sigma(t_1)}{\partial t_1} \frac{\partial f_1(\Theta, x)}{\partial \Theta} \quad (9)$$

$$= \frac{\partial \mathcal{L}}{\partial \hat{y}'} \left(\sigma(W) \frac{\partial \sigma(t_1)}{\partial t_1} \right) \frac{\partial f_1(\Theta, x)}{\partial \Theta}. \quad (10)$$

Comparing (8) with (10) reveals a few crucial points. First of all, in each case the gradient of the loss $\partial \mathcal{L}/\partial \hat{y}'$ is "denoised" by an operator before being fed into the gradient of the base model parameters. This is the main role of the noise model: to prevent the base model from learning the noisy labels directly. In the case of NNAQC, the denoising operator is $\frac{\partial \sigma(t_2)}{\partial t_2} W \frac{\partial \sigma(t_1)}{\partial t_1}$, and in the case of the linear noise model, the denoising operator is $\sigma(W) \frac{\partial \sigma(t_1)}{\partial t_1}$.

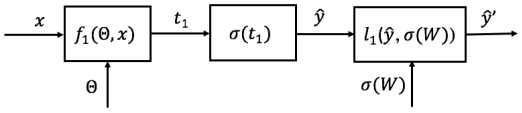

Figure 4: No softmax augmentation

Second, we find that the NNAQC denoising operator is more diffuse than the linear noise model. To see this, consider an estimate $\hat{y}$ that places most of the probability on a single class. In NNAQC, the resulting noisy label prediction is $\hat{y}' = \sigma(W\hat{y})$; applying the softmax to $W\hat{y}$ "spreads out" the probabilities, and the prediction of the noisy label will be less concentrated on a single class than the equivalent linear model. In other words, the nonlinear noise model is intrinsically less confident than a rigorous linear model. Because the denoising operator contains the term $\frac{\partial \sigma(t_2)}{\partial t_2}$, which is a function of $\sigma(W\hat{y})$, the resulting operator is more diffuse, i.e. its columns are less concentrated on individual values.

A more diffuse operator allows for more flexibility in handling disagreements between the CNN model predictions and the noisy labels. Consider the case in which the CNN outputs a prediction $\hat{y}$ concentrated around a single value (say, i) that is different than the (perhaps erroneous) training label y' (say, j). Here, the challenge is to decide whether y' is an error or whether the CNN prediction is bad. In a linear noise model, the denoising operator has most of its weight concentrated on the ith row. On the other hand, the loss gradient $\partial \mathcal{L}/\partial \hat{y}'$ has all of its weight on the jth row. Therefore, the denoising operator wipes out most of the gradient, and the result is largely to ignore the sample. With NNAQC, the denoising operator is not as concentrated around row i, so the backpropagation step attempts to learn more from the training point, even though the model prediction and noisy label disagree.

Similarly, NNAQC prevents the model from being overconfident when the model and noisy label agree. If the CNN makes a confident prediction $\hat{y}$ and $i = j$, the combination of a non-diffuse denoising operator and the gradient $\partial \mathcal{L}/\partial \hat{y}'$ has large-magnitude elements, and the model is overconfident is supposing that the label is not noisy. The diffuse denoising operator resulting from NNAQC, on the other hand, spreads out the gradient, preventing an overaggressive backpropagation step. To sum up, the NNAQC denoising operator encourages the CNN to learn from a training sample when there is disagreement, and discourages overfitting when there is agreement. Finally, the quasi-clustering regularization in our approach in Fig. 2 provides information to base model about the true labels by clustering all the samples that are close in feature space. We also write the backpropagation gradient steps with quasi-clustering regularization as $\frac{\partial \mathcal{L}}{\partial \Theta} = \gamma \frac{\partial \sigma(t_1)}{\partial t_1} \frac{\partial f_1(\Theta, x)}{\partial \Theta}$ where, $\gamma = \left(\frac{\partial \mathcal{L}}{\partial \hat{y}'} \frac{\partial \sigma(t_2)}{\partial t_2} W \right) + \frac{\partial \mathcal{L}'_{QC}}{\partial \hat{y}}$.

We also consider the learning performance when the noise model ϕ is known exactly. One might expect that learning a base CNN using a linear noise model, with the transition matrix set at ϕ, would provide superior performance. Somewhat surprisingly, [10] reports cases where even this "genie-aided" approach is outperformed. We observe the similar behavior; augmenting the CNN with the true noise model performs significantly worse than NNAQC. As suggested by the above analysis, the nonlinear noise model simply results in a more effective denoising operator, even when the model does not learn the underlying noise statistics.

4. Experimental Results

4.1. General Setting

In all the experiments, we use the MATLAB toolbox MatConvNet [30]. We evaluate the performance of the classifier on the MNIST [16], CIFAR-10, CIFAR-100 [12], ImageNet [6] and Clothing1M [33] datasets. We use the default CNN architectures and parameters provided in MatConvNet for CIFAR-10 and MNIST datasets as a base CNN model. For other datasets, we use the CNN architectures that provide the best classification accuracy on corresponding clean datasets and use it as a base CNN. We provide the details of these architectures in subsequent sections. For all

	Training samples	50K						30K						10K				
	Noise %	0	5	10	30	50	70	0	5	10	30	50	70	0	5	10	30	50
CIFAR-10	Base model	**20.49**	23.00	25.30	30.49	39.47	65.60	22.73	24.92	27.63	35.09	45.51	70.30	29.10	30.98	33.94	43.18	52.54
	Genie-aided	20.50	21.07	24.32	28.09	39.29	62.38	22.30	23.16	23.77	30.96	40.18	75.51	27.57	29.05	29.17	41.87	47.02
	Genie-aided (QC)	20.98	22.22	23.23	25.52	33.98	57.57	21.91	23.59	24.37	30.40	39.63	69.09	28.23	29.23	29.80	41.50	46.71
	Trace	22.48	23.00	23.90	27.20	39.06	63.00	23.80	24.50	26.23	30.51	44.60	70.00	31.26	31.71	33.20	41.10	50.21
	Bootstraping	23.33	23.76	25.00	28.64	35.07	66.14	24.50	25.08	26.41	30.68	36.89	61.64	34.00	31.85	33.40	38.27	46.21
	Dropout	37.29	36.90	31.30	25.40	31.28	63.04	44.67	42.73	36.24	33.13	34.50	66.70	43.64	37.43	40.20	41.08	43.64
	F-correction	21.00	21.45	22.10	23.70	29.12	58.91	23.40	23.52	24.12	26.35	32.65	61.20	31.09	32.11	35.00	42.28	44.41
ours	NNAQC	21.11	21.85	**22.03**	24.20	28.41	56.12	**22.72**	**22.91**	**23.01**	25.27	29.48	**56.56**	29.53	29.92	31.02	33.39	37.92
	Regu. NNAQC	20.96	**21.40**	22.05	**23.10**	**28.06**	**56.09**	23.00	23.14	23.58	**25.21**	**29.40**	56.9	**29.00**	**29.06**	**30.21**	**31.86**	**37.80**
	Training Samples	60K						40K						20K				
	Noise %	0	5	10	30	50	70	0	5	10	30	50	70	0	5	10	30	50
MNIST	Base model	**00.89**	02.67	03.68	04.50	34.50	48.80	**01.18**	03.62	06.33	14.78	39.47	56.93	**02.42**	04.86	07.86	21.68	43.20
	Genie-aided	00.89	02.67	03.68	04.50	34.50	48.80	01.48	01.61	02.72	04.37	12.33	50.40	02.39	03.15	04.01	8.47	21.63
	Trace	01.29	01.40	01.46	02.12	03.80	24.20	01.50	01.67	02.00	02.80	04.10	26.32	03.32	03.40	03.89	04.93	08.02
	Bootstraping	01.29	01.30	01.41	02.00	03.60	22.20	01.59	01.77	02.07	02.97	04.19	30.10	03.44	03.67	04.02	05.36	08.59
	Dropout	01.29	01.29	01.32	01.83	02.83	24.60	01.51	01.54	01.90	02.45	03.90	38.07	02.69	03.03	03.39	04.75	06.13
	F-correction	01.12	01.13	01.19	01.50	02.23	21.00	01.42	01.45	01.60	02.10	02.91	23.07	03.17	03.23	03.53	04.46	06.11
ours	NNAQC	01.14	01.15	01.24	01.83	02.20	**16.42**	01.49	01.50	01.51	02.19	03.01	**17.65**	03.00	03.21	03.30	03.57	06.32
	Regu. NNAQC	01.01	**01.08**	**01.18**	**01.46**	**02.19**	18.70	01.36	**01.37**	**01.40**	**02.09**	**02.80**	21.58	03.02	**03.04**	**03.14**	**03.31**	**05.52**

Table 1: NNAQC performance for different datasets and compared to other approaches w.r.t number of training samples.

the NNAQC experiments we use $\alpha = 0.9$ and $\beta = 0.9$. We compare NNAQC to several other algorithms: a standard, noise-ignorant CNN trained on $\mathcal{D}'$ ("base model"); a CNN augmented with the true noise model ϕ ("genie-aided"); the genie-aided model using the quasi-clustering loss function ("genie-aided with QC"); the dropout-regularized model of [10] ("dropout"); the trace-regularized model of [27] ("trace"); the soft bootstrapping algorithm of [26] ("bootstrapping"); and the forward loss correction of [24] ("F-correction"). We also try adding dropout regularization to the noise model of NNAQC ("regu.NNAQC"). For an apples-to-apples comparison we fixed the base model for all the approaches and implement their methods on top of it. In all the experiments we train CNN end-to-end via stochastic gradient descent method with batch size 100. For CIFAR-10 and MNIST datasets, we run the experiment 5 times for each setting and report the mean.

4.2. Artificial Label Noise

To examine the robustness of NNAQC on artificially injected noise, we corrupt the true labels according to (1) with $p \in \{0, 0.05, 0.10, 0.30, 0.50, 0.70\}$.

CIFAR-10: We train our CNN on CIFAR-10 dataset [12], a subset of 80 million Tiny Image dataset [29]. It contains natural images of size $32 \times 32 \times 3$ from 10 different categories. It has $50K$ training and $10K$ test images. On the *clean* dataset, the base model CNN achieves 20.49% classification error. We produce a noisy dataset $\mathcal{D}'$ by corrupting the labels according to the noise distribution (1) for each value of p. Table 1 first row shows the comparative performance of NNAQC when the networks are trained using 50K, 30K, and 10K training samples, respectively. In all cases, NNAQC, perhaps regularized by dropout, substantially outperforms other approaches. This includes the genie-aided approaches, bolstering our claim that it is less important to know the noise statistics than to learn an effective denoising operator for training. Further, NNAQC is robust to variations in the noise level, recovering near-optimum performance when there is little noise. Although, we notice that NNAQC performances better than NNAQC with dropout regularization (Regu. NNAQC) in some of the cases, but this performance gap is negligible. However, we observe a significant performance gap with the datasets having more than 10 classes.

To evaluate the robustness of NNAQC with respect to varying training dataset size, in Table 1, we show the performance of all the approaches as a function of number of training samples. For every dataset, we starts with original number of training samples and keep on decrease the samples by 20K, as shown in Table 1. For CIFAR-10 dataset, in column 1 we train all the models with all 50K training samples, in column 2 we train all the models with 30K (column 2) training samples and so on. We observe that in all the noise regimes, when the training sample size is decreasing from 50K to 30K to 10K, change in NNAQC performance is very small, while the performance of other approaches shows a high performance gap, therefore, depends on amount of training samples. We also find that the performance of *F-correction* [24] is close to the performance to the NNAQC, however, as we reduce the training dataset size NNAQC outperforms F-correction significantly. Since [24] works by estimating the noise transition matrix, the performance gap on smaller training set further strengthens our claim that learning a correct noise model is neither necessary nor sufficient for state-of-the-art performance in the presence of label noise.

We also compare NNAQC to [3], which uses a pre-trained AlexNet to obtain high level features for training images and fine-tunes a final softmax layer on $\mathcal{D}'$. Because

they use a pre-trained network where NNAQC and other approaches train a CNN from scratch, a direct comparison of results is impossible. However, in the presence of 50% noise for $50K$ training samples, [3] reports 28% classification error rate, compared to 28.41% for NNAQC. That is, NNAQC performs competitively with this approach even though it is not pre-trained, which may indicate that it is a more powerful approach overall.

MNIST: We perform similar experiments on handwritten digits dataset MNIST [16], which contains $60K$ training images of the 10 digits of size 28×28 and $10K$ test images. We produce a noisy dataset $\mathcal{D}'$ as in the CIFAR-10 case. On the clean dataset, the base model CNN achieves a classification error rate of 0.89%. In Table 1 (Last row) we again see that NNAQC provides superior performance overall and is robust to both high noise power and a smaller training set.

Similar to CIFAR-10, we compare the performance of NNAQC against the pre-trained /fine-tuned strategy [3] on the MNIST dataset. In the presence of 50% noise NNAQC outperforms the [3], achieving 2.2% classification error while [3] achieves at minimum 7.63% classification error.

CIFAR-100: We next show the performance of NNAQC on a dataset with more classes, making the problem more challenging: CIFAR-100 [12] which consists of 32×32 color images of 100 different categories containing 600 images each. There are 500 images for training and 100 images for testing per class. Because of the complexity of this datatset, we use a different base CNN model with two conv+ReLU+max pool layers, two FC layers and a softmax layer. This is a low capacity CNN network (LC-CNN) with a classification error rate of 50.9% on the clean dataset. In order to verify that the robustness of NNAQC is not due to the low capacity models, we also evaluate NNAQC on a high capacity deep residual network (ResNet) [9] with 30.99% classification error rate on clean labels. We use ResNet with depth 44 and the same training parameters as described in [24].

We compare the NNAQC performance on CIFAR-100 in Table 2. Here we train the networks on entire training data. Similar to previous experiments we fixed the base model CNN for all the approaches. In Table 2 (First row), we show the competitive performance of NNAQC over other approaches when trained on LC-CNN. We observed that the performance of NNAQC on CIFAR-100 is consistent with MNIST and CIFAR-10, proves the scalability of NNAQC. Here, dropout particularly improves performance (Regu. NNAQC), likely because the larger label noise model benefits from regularization.

We also show the performance of NNAQC on ResNet architecture in Table 2 (Second row). We observe that among other approaches only F-correction performs equally well with NNAQC at a number of occasions, however, with

	Noise %	0	5	10	30	50	60
LC-CNN	Base model	**50.90**	52.48	53.82	60.38	68.46	88.20
	Trace	53.12	54.27	55.00	58.70	64.50	84.12
	Bootstraping	54.20	54.90	55.30	59.00	69.75	88.30
	Dropout	65.80	63.54	62.01	57.76	63.24	84.19
	F-correction	56.68	57.13	57.11	62.67	66.12	83.90
ours	NNAQC	52.31	52.40	53.10	56.68	63.00	84.00
	Regu. NNAQC	52.29	**52.33**	**53.00**	**56.91**	**62.20**	**83.13**
	Noise %	0	5	10	30	50	60
44-layer ResNet	Base model	**30.99**	31.54	33.86	36.50	64.60	84.89
	Trace	31.56	31.50	34.10	36.00	65.41	84.82
	Bootstraping	31.60	31.50	34.06	36.32	63.45	84.30
	Dropout	55.20	53.04	52.13	37.68	64.11	85.00
	F-correction	31.00	31.13	**33.12**	35.80	61.24	**84.00**
ours	NNAQC	31.00	31.14	34.01	35.88	61.35	85.00
	Regu. NNAQC	31.12	**31.13**	33.16	**35.71**	**61.20**	84.03

Table 2: NNAQC performance on CIFAR-100 with different CNN architectures and compared to other approaches.

the LC-CNN the scenario is different—NNAQC performs better than all the other approaches. Comparing NNAQC performance on ResNet with LC-CNN, it is clear that the NNAQC performance is independent of base CNN network architecture. This claim is further strengthened by our experiments on Clothing 1M datasets with different CNN architectures in the next section.

ImageNet: We further test the scalability of NNAQC to a 1000 class classification problem. We show the performance of NNAQC on ImageNet 2012 dataset [6] which has 1.3M image with clean labels over 1000 categories. For this experiment, we use CNN model of Krizhevsky et al. [13] as the base model. This CNN model has five conv+RELU+max pool layers, two FC layers and a softmax layer. As described in [27], we generate a column stochastic noise distribution matrix (ϕ) such that for a particular class, noise is randomly distributed to only 10 other randomly chosen classes. For 50% label noise, each class has 50% correct labels and other 50% labels are randomly distributed among 10 randomly chosen classes. Since our

		Top 5 Val. error		
ImageNet	Noise %	0	10	50
	Base model	19.20	31.21	53.46
	Trace	19.10	29.00	46.24
ours	NNAQC	19.30	29.10	44.31
	Regu. NNAQC	**18.30**	**28.21**	**41.80**

Table 3: ImageNet validation set classification error rate.

main intention here is to show the scalability of NNAQC to a large number of classes and to maintain the simplicity, we transfer the parameters of first four convolutional blocks from a pre-trainined AlexNet model. While training, we keep the parameters of first four convolutional blocks (conv+ReLU+max pool) intact / frozen and only train the last convolutional block, two FC layers, a softmax layer and the stacked NNAQC layer. In Table 3 we compare the NNAQC performance with the base Alexnet model (i.e., no

noise model) on the validation set images, with 0%, 10%, and 50%, randomly-distributed corrupted labels. We observe a slight performance gain for NNAQC over the base model with clean labels-perhaps due to label noise inherent in the ImageNet dataset. We observe that the 50% label noise significantly hurts the performance of the base model whereas the NNAQC withstands and shows a superior performance (a clear gain of $\sim$ 11.0%) over the base model. Here, dropout regularization (Regu. NNAQC) further improves the overall performance by 2.51%.

4.3. Real Label Noise

Finally, we evaluate the performance of NNAQC on real world noisy label dataset Clothing 1M [33] in terms of classification error rate. This dataset contains 1M images with noisy labels from 14 different classes. Along with the incorrectly labeled images, this dataset provides 50K clean images for training; 14k for validation; and 10k for testing. For this dataset, we use a 50-layer ResNet pre-trained on ImageNet dataset as a base model. Similar to [24], we train the network with different weight-decay parameter depending on the training dataset size. In Table 4 we compare the performance of NNAQC with a number of existing approaches.

Clothing 1M				
#	model /method	init	training	error
1	AlexNet /cross-	ImageNet	50k	28.17
2	AlexNet /trace	#1	1M,50k	24.84
3	50-ResNet /cross-	ImageNet	50K	25.12
4	50-ResNet /F-corr-	ImageNet	1M	30.16
5	50-ResNet /cross-	#4	50k	19.62
6	50-ResNet /[28]	ImageNet	1M	27.77
7	50-ResNet /NNAQC	ImageNet	50K	25.10
8	50-ResNet /NNAQC	ImageNet	1M	27.73
9	50-ResNet /NNAQC	ImageNet	1M, 50K	24.58
10	50-ResNet /cross-	#8	50K	**19.45**

Table 4: NNAQC performance on clothing1M dataset. #10 shows the best results. #6 is reported results from [28] and cross- represents the cross entropy loss.

At first, we see a clear performance improvement of $\sim$ 3% with ResNet in comparison to AlexNet (#1 vs #3). On clean training images NNAQC (#7) performs better than the base model (#3) as expected. On noisy images with ImageNet pretraining, we gain a 3% performance improvement compared to F-correction. Also, in comparison to a very recent work [28] (#6 vs #8), NNAQC performance is very competitive. Further, we observe the effect of availability of clean 50k images on the NNAQC performance, that is, given the clean labels, NNAQC performance improved by $\sim$ 3% (#8 vs #9). In a similar vein to [24], we first train NNAQC on 1M noisy images (#8) and fine tune the network with 50k clean images (#10), we observe that in comparison to #5, the NNAQC outperforms all the methods in Table 4 and is very competitive overall.

4.4. Effect of Different Components

We perform an ablative study to observe the contribution of individual components, the non-linear noise aware component (NNA) and the quasi-clustering component (QC) on the overall performance of NNAQC. To observe the effect of non-Linear noise aware component, we train the NNAQC with $\alpha = 1$ in (6) and to observe the individual effect of quasi-clustering component, we set $\alpha = 0$ in (6).

Dataset	Loss	Label Noise					
		0	5	10	30	50	70
CIFAR-10	NNA	36.79	36.10	33.03	26.43	31.11	59.65
	QC	23.32	23.57	25.09	28.64	35.07	65.20
	NNAQC	**21.11**	**21.85**	**22.03**	**24.20**	**28.41**	**56.92**
MNIST	NNA	01.30	01.30	01.38	01.91	03.01	23.22
	QC	01.34	01.42	01.61	02.27	04.15	26.10
	NNAQC	**01.01**	**01.08**	**01.18**	**01.46**	**02.19**	**18.70**
CIFAR-100	NNA	63.01	62.43	60.01	58.15	62.40	–
	QC	54.20	54.90	55.30	59.00	69.75	–
	NNAQC	**52.29**	**52.33**	**53.00**	**56.91**	**62.20**	–

Table 5: Effect of NNA and QC component on the overall performance of NNAQC

In Table 5 we find that the individual components did not perform well, whereas the combination of these components results in the state-of-the-art performance. For instance, when NNAQC is trained on CIFAR-10 with 50% label noise, we achieve an overall classification error rate of 28.06%. Now, if the QC loss component is removed (or $\alpha = 1$), the classification error rate jumps to 31.11%. Also, when only the non-linear noise aware (NNA) component is removed, that is $\alpha = 0$, the classification error rate jumps to 35.07%. We observe the similar behavior with MNIST and CIFAR-100 as well. Therefore, both the components contribute to the overall performance of NNAQC. We cross-validate the hyperparameter α on the set of noisy labeled images and find that a value of $\alpha = 0.9$ works best for a majority of the experiments on different datasets.

5. Conclusion and Future Work

In this paper we propose a scalable and effective nonlinear, noise-aware quasi clustering approach towards training a deep CNN on noisy data labels. We show the performance of NNAQC on variety of different datasets with different noise regimes and varying training data sizes. Further, we anticipate that our model can handle instance dependent label noise as well, that is, QC step accounts for instance-dependent noise without learning a full instance-dependent noise model. Future works shall consider analyzing the instance dependent label noise.

References

[1] J. A. Aslam and S. E. Decatur. On the sample complexity of noise-tolerant learning. *Information Processing Letters*, 57(4):189–195, 1996. 2

[2] K. Audhkhasi, O. Osoba, and B. Kosko. Noise-enhanced convolutional neural networks. *Neural Networks*, 78:15–23, 2016. 4

[3] S. Azadi, J. Feng, S. Jegelka, and T. Darrell. Auxiliary image regularization for deep cnns with noisy labels. *arXiv preprint arXiv:1511.07069*, 2015. 2, 6, 7

[4] C. E. Brodley and M. A. Friedl. Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, 11:131–167, 1999. 2

[5] C. E. Brodley, M. A. Friedl, et al. Identifying and eliminating mislabeled training instances. In *AAAI/IAAI, Vol. 1*, pages 799–805, 1996. 2

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 1, 5, 7

[7] B. Frénay and M. Verleysen. Classification in the presence of label noise: a survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, 2014. 1

[8] J. Goldberger and E. Ben-Reuven. Training deep neural-networks using a noise adaptation layer. 2017. 4

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 7

[10] I. Jindal, M. Nokleby, and X. Chen. Learning deep networks from noisy labels with dropout regularization. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 967–972. IEEE, 2016. 2, 3, 5, 6

[11] I. Jindal, D. Pressel, B. Lester, and M. Nokleby. An effective label noise model for dnn text classification. *arXiv preprint arXiv:1903.07507*, 2019. 2

[12] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 1, 5, 6, 7

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1, 7

[14] N. D. Lawrence and B. Schölkopf. Estimating a kernel fisher discriminant in the presence of label noise. In *ICML*, volume 1, pages 306–313. Citeseer, 2001. 2

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[16] Y. LeCun, C. Cortes, and C. J. Burges. The mnist database of handwritten digits, 1998. 5, 7

[17] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li. Learning from noisy labels with distillation. In *ICCV*, pages 1928–1936, 2017. 2

[18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014. 1

[19] T. Liu and D. Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on pattern analysis and machine intelligence*, 38(3):447–461, 2016. 2

[20] X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. M. Erfani, S.-T. Xia, S. Wijewickrema, and J. Bailey. Dimensionality-driven learning with noisy labels. *arXiv preprint arXiv:1806.02612*, 2018. 2

[21] E. Malach and S. Shalev-Shwartz. Decoupling" when to update" from" how to update". In *Advances in Neural Information Processing Systems*, pages 961–971, 2017. 2

[22] N. Manwani and P. Sastry. Noise tolerance under risk minimization. *IEEE transactions on cybernetics*, 43(3):1146–1151, 2013. 2

[23] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari. Learning with noisy labels. In *Advances in Neural Information Processing Systems*, pages 1196–1204, 2013. 2

[24] G. Patrini, A. Rozza, A. K. Menon, R. Nock, and L. Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.(CVPR)*, pages 2233–2241, 2017. 2, 6, 7, 8

[25] U. Rebbapragada and C. E. Brodley. Class noise mitigation through instance weighting. In *European Conference on Machine Learning*, pages 708–715. Springer, 2007. 2

[26] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014. 2, 4, 6

[27] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014. 2, 3, 4, 6, 7

[28] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa. Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5552–5560, 2018. 2, 8

[29] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–1970, 2008. 1, 6

[30] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for matlab. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 689–692. ACM, 2015. 5

[31] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie. Learning from noisy large-scale datasets with minimal supervision. In *The Conference on Computer Vision and Pattern Recognition*, 2017. 2

[32] Y. Wang, W. Liu, X. Ma, J. Bailey, H. Zha, L. Song, and S.-T. Xia. Iterative learning with open-set noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8688–8696, 2018. 2

[33] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. 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. 2, 5, 8

[34] X. Zhu. Semi-supervised learning literature survey. 2005. 2