Training Convolutional Networks with Noisy Labels

Sainbayar Sukhbaatar[1], Joan Bruna[2], Manohar Paluri[2], Lubomir Bourdev[2] & Rob Fergus[2]
[1]Dept. of Computer Science, Courant Institute, New York University [2]Facebook AI Research

Abstract

The availability of large labeled datasets has allowed Convolutional Network models to achieve impressive recognition results [1]. However, in many settings manual annotation of the data is impractical; instead our data has *noisy* labels, i.e. there is some freely available label for each image which may or may not be accurate [2, 3, 4]. In this paper, we explore the performance of discriminatively-trained Convnets when trained on such noisy data. We introduce an extra noise layer into the network which adapts the network outputs to match the noisy label distribution. The parameters of this noise layer can be estimated as part of the training process and involve simple modifications to current training infrastructures for deep networks.

Label Flip Noise

Given training data $(\mathbf{x}_n, y_n^*)$ where y^* denotes the true labels $\in 1, \dots K$, we define a noisy label distribution $\tilde{y}$ given by $p(\tilde{y} = j|y^* = i) = q_{j,i}^*$, parametrized by a $K \times K$ probability transition matrix $Q^* = (q_{ji}^*)$. We thus assume here that label flips are independent of $\mathbf{x}$. The probability that an input $\mathbf{x}$ is labeled as j in the noisy data can be computed using Q^*

$$p(\tilde{y} = j|\mathbf{x}) = \sum_i p(\tilde{y} = j|y^* = i)p(y^* = i|\mathbf{x}) = \sum_i q_{ji}^* p(y^* = i|\mathbf{x}). \quad (1)$$

In the same way, we can modify a classification model using a probability matrix Q that modifies its prediction to match the label distribution of the noisy data. Let $\hat{p}(y^*|\mathbf{x}, \theta)$ be the prediction probability of true labels by the classification model. Then, the prediction of the combined model will be given by

$$\hat{p}(\tilde{y} = j|\mathbf{x}, \theta, Q) = \sum_i q_{ji} \hat{p}(y^* = i|\mathbf{x}, \theta). \quad (2)$$

This combined model is trained by maximizing the cross entropy between the noisy labels $\tilde{y}$ and the model prediction given by Eqn. 2. The cost function to minimize is

$$\mathcal{L}(\theta, Q) = -\frac{1}{N} \sum_{n=1}^{N} \log \hat{p}(\tilde{y} = \tilde{y}_n|\mathbf{x}_n, \theta, Q) = -\frac{1}{N} \sum_{n=1}^{N} \log \left(\sum_i q_{\tilde{y}_n i} \hat{p}(y^* = i|\mathbf{x}_n, \theta) \right). \quad (3)$$

However, the ultimate goal is to predict true labels y^*, not the noisy labels $\tilde{y}$. This can be achieved if we can make the base model predict the true labels accurately. One way to quantify this is to use its confusion matrix $C = \{c_{ij}\}$ defined by

$$c_{ij} := \frac{1}{|S_j|} \sum_{n \in S_j} \hat{p}(y^* = i|\mathbf{x}_n, \theta), \quad (4)$$

where S_j is the set of training samples that have true label $y^* = j$. If we manage to make C equal to identity, that means the base model perfectly predicts the true labels in training data. We can also define the confusion matrix $\tilde{C} = \{\tilde{c}_{ij}\}$ for the combined model in the same way

$$\tilde{c}_{ij} := \frac{1}{|S_j|} \sum_{n \in S_j} \hat{p}(\tilde{y} = i|\mathbf{x}_n, \theta, Q). \quad (5)$$

Using Eqn. 2, it follows that $\tilde{C} = QC$. Let us show that minimizing the training objective in Eqn. 3 forces the predicted distribution from the combined model to be as close as possible to the noisy label distribution of the training data, asymptotically. As $N \to \infty$, the objective in Eqn. 3 becomes

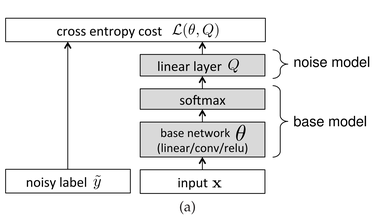
$$\mathcal{L}(\theta, Q) = -\frac{1}{N} \sum_{n=1}^{N} \log \hat{p}(\tilde{y} = \tilde{y}_n|\mathbf{x}_n) = -\frac{1}{N} \sum_{k=1}^{K} \sum_{n \in S_k} \log \hat{p}(\tilde{y} = \tilde{y}_n|\mathbf{x}_n, y_n^* = k)$$
$$\xrightarrow{N \to \infty} -\sum_{k=1}^{K} \sum_{i=1}^{K} q_{ik}^* \log \hat{p}(\tilde{y} = i|\mathbf{x}, y^* = k) \geq -\sum_{k=1}^{K} \sum_{i=1}^{K} q_{ik}^* \log q_{ik}^*, \quad (6)$$

since $-\sum_k q_k^* \log \hat{p}_k \geq -\sum_k q_k^* \log q_k^* = H(q^*)$, and with equality in the last equation only when $\hat{p}(\tilde{y} = i|\mathbf{x}, y^* = k) = q_{ik}^*$. In other words, the model tries to match the confusion matrix $\tilde{C}$ of the combined model to the true noise distribution Q^* of the noisy data

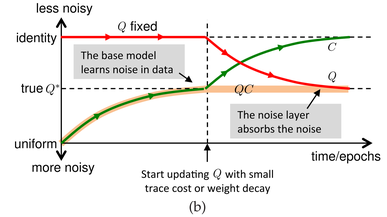
$$\tilde{c}_{ik} = 1/|S_k| \sum_{n \in S_k} \hat{p}(\tilde{y} = i|\mathbf{x}_n, y_n^* = k) \to q_{ik}^* \implies \tilde{C} = QC \to Q^*. \quad (7)$$

If we know the true noise distribution Q^* and it is non-singular, then from Eqn. 7, setting $Q = Q^*$ would force C to converge to identity. Therefore, training to predict the noisy labels using the combined model parameterized by Q^* directly forces the base model to predict the true labels.

Learning The Noise Distribution



(a)



(b)

Figure 1: (a) Label noise is modeled by a constrained linear layer inserted between softmax and cost layers. **(b)** The training sequence when learning from noisy data.

In practice, the true noise distribution Q^* is often unknown to us. In this case, we have to infer it from the noisy data itself. Fortunately, the noise model is a constrained linear layer in our network, which means its weights Q can be updated along with other weights in the network. This is done by back-propagating the cross-entropy loss through the Q matrix, down into the base model. After taking a gradient step with the Q and the model weights, we project Q back to the subspace of probability matrices because it represents conditional probabilities.

Unfortunately, simply minimizing the loss in Eqn. 3 will not give us the desired solution. From (6), it follows that $QC = \tilde{C} \to Q^*$ as the training progresses, where $\tilde{C}$ is the confusion matrix of the combined model and Q^* is the true noise distribution of data. However, this alone cannot guarantee $Q \to Q^*$ and $C \to I_K$.

In order to force $Q \to Q^*$, we add a regularizer on the probability matrix Q which forces it to diffuse, such as a trace norm or a ridge regression. This regularizer effectively transfers the label noise distribution from the base model to the noise model, encouraging Q to converge to Q^*.

Outlier Noise

Another important setting is the case where some training samples do not belong to any of the existing signal classes. In that case, we can create an additional "outlier" class, which enables us to apply the previously described noise model.

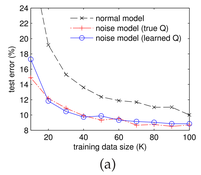
Let K be the number of the existing classes. Then, the base network should output $K + 1$ probabilities now, where the last one represents the probability of a sample being an outlier. If the labels given to outlier samples are uniformly distributed across classes, then the corresponding noise distribution Q^* becomes a $K + 1 \times K + 1$ matrix

$$Q^* = \begin{pmatrix} 1 & 0 & \cdots & 0 & 1/K \\ 0 & 1 & \cdots & 0 & 1/K \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 1/K \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \to \begin{pmatrix} 1 & 0 & \cdots & 0 & (1-\alpha)/K \\ 0 & 1 & \cdots & 0 & (1-\alpha)/K \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & (1-\alpha)/K \\ 0 & 0 & \cdots & 0 & \alpha \end{pmatrix}. \quad (8)$$

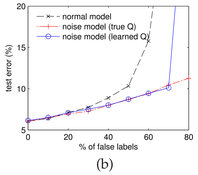
Unfortunately, this matrix is singular. A simple solution to this problem is to add some extra outlier images with label $K + 1$ in the training data, which would make Q^* non-singular (in most cases, it is cheap to obtain such extra outlier samples).

Experiments

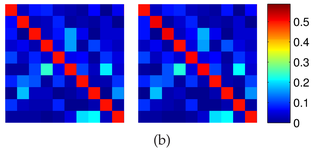
Label Flip Noise



(a)



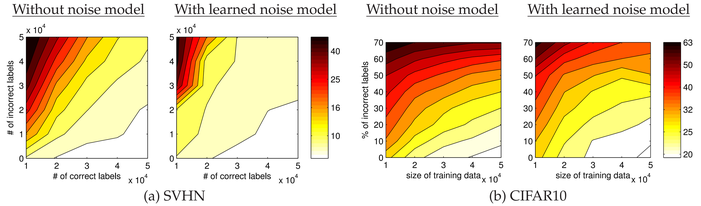
(b)



(b)

Figure 2: (a) Test errors on SVHN dataset when the noise level is 50% for differing overall training set sizes. (b) Test errors when trained on 100k samples, as the noise level varies. Note that the performance for learned Q is very close to a model trained with Q fixed to the true noise distribution Q^*. (c) The ground truth noise distribution Q^* (left) and Q learned from noisy data (right).

Experiments (continued)



(a) SVHN (b) CIFAR10

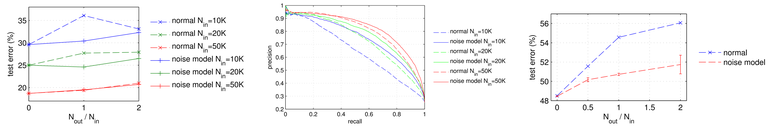
Figure 3: The noise model is compared to the baseline model on different amount of training data and varying noise levels. The plots show test errors (%), where brighter color indicates better accuracy.

Noise model	Training size	Noise %	Valid. Error
None	1.2M	0	39.8%
None	0.6M	0	48.5%
None	1.2M	50	53.7%
Learned Q	1.2M	50	45.2%
True Q^*	1.2M	50	41.4%

Noise model	Training size	Noise %	Valid. error
None	1.2M	40	50.5%
Learned Q	1.2M	40	46.7%
True Q^*	1.2M	40	43.3%

Table 1: Effect of label flip noise using the ImageNet dataset

Outlier Noise



(a) (b) (c)

Figure 4: (a) The effect of the outlier noise on the classification performance on CIFAR10 with and without the noise model. Here, N_{in} and N_{out} are the number of inlier and outlier images in the training data, respectively. (b) Precision recall curve for detecting inliers in test data. (c) ImageNet outlier experiments with varying ratios of outlier/inliers.

Real Label Noise

Method	Valid. error
Normal Convnet	48.8%
Label-flip model	48.2%
Outlier model	48.5%

Table 2: Evaluation on our real-world Web image + ImageNet noisy dataset.

Conclusion

In this paper we explored how convolutional networks can be trained on data with noisy labels. We proposed two simple models for improving noise robustness, focusing different types of noise. We explored both approaches in a variety of settings: small and large-scale datasets, as well as synthesized and real label noise. In the former case, both approaches gave significant performance gains over a standard model. On real data, then gains were smaller. However, both approaches can be implemented with minimal effort in existing deep learning implementations, so add little overhead to any training procedure.

References

[1] Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. (2012)
[2] Mnih, V., Hinton, G.: Learning to label aerial images from noisy data. (2012)
[3] Bootkrajang, J., Kabn, A.: Label-noise robust logistic regression and its applications. (2012)
[4] Natarajan, N., Dhillon, Inderjit, Ravikumar, Pradeep, and Tewari, A.: Learning with noisy labels (2013)