Training Deep Neural-Networks using a Noise Adaptation Layer

Jacob Goldberger Ehud Ben-Reuven

Bar-Ilan University, Israel

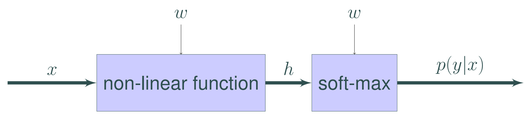
Introduction

- The presence of inaccurate class labels is known to deteriorate the performance of even the best classifiers.
- Noisy labels tend to be more harmful than noisy features.
- We present a training procedure that is suitable for noisy labels.

Network Modeling

Standard network with softmax output layer (baseline model):

$$p(y=i|x;w)=\frac{\exp(u_i^\top h(x)+b_i)}{\sum_l \exp(u_l^\top h(x)+b_l)}$$



We next add (only at training phase) another softmax output layer to predict the noisy label z based on both the true label y and the input features x (c-model):

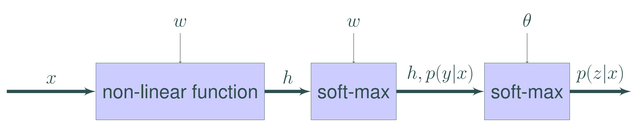
$$p(z=j|y=i,x)=\frac{\exp(u_{ij}^\top h(x)+b_{ij})}{\sum_l \exp(u_{il}^\top h(x)+b_{il})}$$

$$p(z=j|x)=\sum_i p(z=j|y=i,x)p(y=i|x)$$

We can also define a simplified version (s-model) where the noisy label only depends on the true label:

$$p(z=j|y=i)=\frac{\exp(b_{ij})}{\sum_l \exp(b_{il})}$$

$$p(z=j|x)=\sum_i p(z=j|y=i)p(y=i|x)$$



Network training

We are given training data $x_1,...,x_n$ with noisy labels $z_1,....,z_n$. The true labels $y_1,...,y_n$ are hidden random variables.

Likelihood: $L(w,\theta)=\sum_t \log(\sum_i p(z_t|y_t=i,x_t;w,\theta)p(y_t=i|x_t;w))$

EM Approach for s-model (when noise depends only on the labels)

E-step:

$$c_{ti}=p(y_t=i|x_t,z_t)\propto p(y_t=i|x_t;w)p(z_t|y_t=i;\theta)$$

M-step: train a NN by optimizing the objective function:

$$S(w)=\sum_t\sum_i c_{ti}\log p(y_t=i|x_t;w)$$

- EM is a greedy optimization procedure.
- We need to train a NN at each iteration.
- The EM approach is not scalable.
- EM is not applicable when the noise depends also on the features.

Direct Approach

Jointly train the network and the noise adaptation layer to maximize the likelihood:

$$L(w,\theta)=\sum_t \log(\sum_i p(z_t|y_t=i,x_t;w,\theta)p(y_t=i|x_t;w))$$

$$\frac{\partial L}{\partial\theta}=\sum_t\sum_i p(y_t=i|x_t,z_t)\cdot\frac{\partial}{\partial\theta}\log p(z_t|y_t=i,x_t;w,\theta)$$

Parameter Initialization

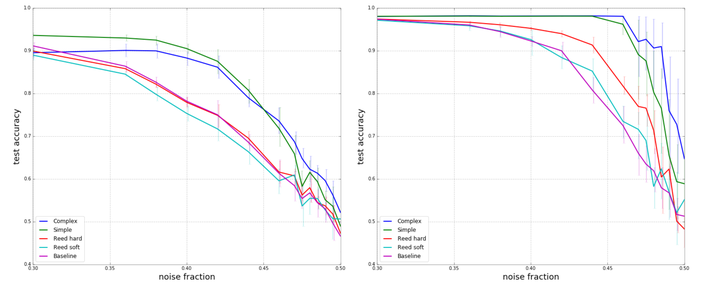
- Use the parameters of the original network to initialize the parameters of the s-model network that contains the noise adaptation level.
- We can then treat the labels produced by the original NN as the true labels. Compute the confusion matrix on the train set and used it as an initial value for the bias parameters:

$$b_{ij}=\log(\frac{\sum_t 1_{\{z_t=j\}}p(y_t=i|x_t)}{\sum_t p(y_t=i|x_t)})$$

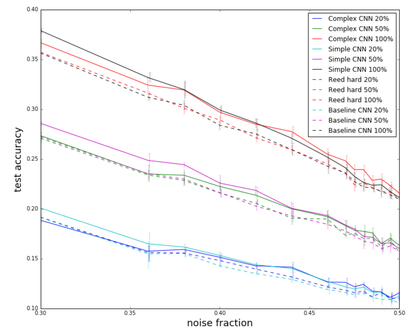
- To initialize the c-model parameters use the bias terms b_{ij} that were optimized for the s-model and set linear terms u_{ij} to zero.

The code and a one line of Keras that implements it:
https://github.com/udibr/noisy_labels

Experimental Results



Test classification accuracy results on the MNIST dataset as a function of the noise level. The results are shown for two training data sizes (20%,100%) of the training subset.



Test classification accuracy results on the CIFAR-100 dataset as a function of the noise level. The results are shown for several training data sizes (20%,50%,100%) of the training subset for a CNN network architecture.

Conclusions

- We proposed an algorithm for training neural networks based solely on noisy data where the noise distribution is unknown.
- We showed that we can reliably learn the noise distribution from the noisy data without using any clean data which, in many cases, are not available.
- Our results encourage collecting more data at a cheaper price, since mistaken data labels can be less harmful to performance.
- One possible future research direction would be to generalize our learning scheme to cases where both the features and the labels are noisy.