Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/43/20230845

Learning noisy transition matrix using a neural network

Yongliang Miao[1,5,†], Kuntian Tang[2,†], Chen Wang[3,†], Yuqi Cao[4,†]

[1]Beijing normal university-HongKong baptist university united international collage, Zhuhai, 519000, China
[2] Guangdong university of Finance and Economics, Guangzhou, 510240, China
[3]South China normal university, Foshan, 528225, China
[4]No.59 Middle School, Taiyuan, 030002, China

[5]924667541@qq.com
[†]These authors contributed equally to this work and should be considered co-second authors.

Abstract. In the field of deep learning, it is crucial to know the accurate distribution of dataset. However, to obtain a high quality of dataset using traditional methods is prone to be both costly and inefficient. Using deep learning methods to estimate the noisy transition matrix provides a feasible way, as the result of its essential function of learning to denote the relationship between clean labels and noisy labels and building a statistically consistent classifier. The major difficulty of learning the noisy transition matrix stems from the unavailability of the distribution of clean data and noisy data. In this paper, we propose a practical and convenient method to study a combination of augumentation and a novel loss function, only leveraging the already known clean labels to aid in learning the noisy transition matrix in the whole dataset. Finally, Through the experiment, the result demonstrates a superior performance and generalization capabilities of the proposed method.

Keywords: noisy transition matrix, label transition neural network, the classifier head.

1. Introduction

In machine learning, data is always the most crucial resource. Nowadays, many large-scale datasets are frequently collected from ranging from crowdsourcing platforms, web crawling and online queries to image engines. These data sources are often subject to unavoidable label noise caused by erroneous annotations. The noisy data disturb the model in terms of Interfering with training, learning erroneous features, increasing uncertainty, and inducing data bias [1]. One way to detect noisy data is through manual annotation. This way is high accuracy but is also time-consuming and money-consuming when the size of dataset is large-scale. To improve annotation efficiency and reduce time costs, the researches on the methods to learn model with noisy labels are increasingly emphasized.

Typically, method for categorizing approaches for addressing noisy labels into two groups: statistically inconsistent methods and statistically consistent methods. Statistically inconsistent methods are heuristic approaches selecting possible clean samples to train the classifier [2]. which reweight examples to mitigate the impact of noisy labels, correcting labels or adding regularization n [3]. While these methods often perform well empirically; There is untheoretical assurance that the acquired classifiers will ultimately converge to the optimal classifiers obtained from clean data.

To overcome this constraint, an alternative approach involves developing algorithms that ensure classifier consistency. These algorithms aim to train classifiers on noisy data in such a way that they eventually converge to the optimal classifiers derived from clean data. The label transition matrix $T(x)$ plays a crucial role in constructing statistically consistent algorithms [4]. (Traditionally, $T(x)$ is defined to relate the clean distribution with the noisy distribution, where $T(x) = P(\tilde{Y} \mid Y,\ X = x)$, with X representing the random variable, $\tilde{Y}$ as the variable for noisy labels, and Y as the variable for clean labels.

In practical situations, we often encounter the challenge of not knowing the clean-label transition matrix, also commonly referred to as T(x). This matrix is instrumental in training a clean label classifier from noisy data. The objective of this classifier is to estimate the probability distribution of clean class labels, denoted as P(Y | X), when given an input. This probability distribution essentially represents the underlying distribution from which clean labels are drawn. However, in real-world scenarios, the clean-label transition matrix is typically unknown.

Inspired by the concept of the transition matrix, we recommend a method which we randomly select a subset from the entire noisy dataset for high-quality annotation, creating a small portion of clean dataset as confident samples. We then train a classifier using these confident samples. To move on, we use the classifier to predict the clean labels for the original noisy dataset and utilize the original labels to train an instance-specific noisy transition matrix. This approach utilizes deep neural networks to estimate an instance-dependent label transition matrix within a reduced feasible solution space, and this neural network is referred to as the noisy transition matrix.

2. Related work

Loss function. There are some popular loss functions used for handling noise in the field of machine learning.

2.1. GCE

This loss function is a novel loss function specifically designed for classification tasks, aiming to address common label noise issues in large-scale datasets. It is an improvement and combination of two commonly used loss functions, Mean Absolute Error (MAE) and Cross-Entropy Loss (CCE), to strike a balance between noise robustness and better learning dynamics [5].

The Lq loss function is proposed, which is a generalization of CCE and MAE. By adjusting the parameter q between 0 and 1, the nature of the loss function can be controlled. Smaller q values make the loss function more robust, enabling better handling of label noise, while larger q values emphasize learning dynamics, facilitating faster convergence.

Furthermore, to further enhance tolerance to noise, the paper introduces the Truncated Lq loss function. In this loss function, a threshold k is introduced to determine which samples should be retained and which samples should be pruned. Only samples with softmax output values higher than k will influence the loss function calculation. Through pruning, the algorithm can focus more on clean data, disregarding the impact of noise labels, thereby improving noise tolerance.

2.2. Joint

The paper about the joint loss function, proposes a label-optimized training method for training classification networks with noisy labels. This method is achieved through alternate optimization of network parameters and labels. Initially, it is assumed that training the network with a high learning rate may lead to difficulties in adapting to noisy labels. To address this, the paper suggests reducing the loss function by updating the labels. Subsequently, a joint optimization problem of network parameters and labels is introduced, accomplished by minimizing a comprehensive loss function comprising classification loss and two regularization terms [6].

Specifically, the paper defines the loss function and outlines the roles of the two regularization terms. The regularization term Lp prevents all labels from being assigned to a single category by constraining the prior probability distribution using KL divergence. The regularization term Le enhances the concentration of the classification distribution when using soft labels, achieved through entropy.

Finally, the paper presents an alternating optimization algorithm to solve the problem. In this algorithm, network parameters and labels are alternately updated, and the label updates can be performed using either hard or soft label methods.

Other methods. There are other more complex training frameworks or processes, including but not limited to robust loss functions, sample selection, label correction, (implicit) regularization, semi-supervised learning, combinations of semi-supervised learning, MixUp, regularization and Gaussian mixture models.

3. Method

3.1. The usage of noisy transition matrix

The noise transition matrix $T(x)$ is commonly used to model the generation of label noise. The transition matrix's individual element $T_{ij}(x)$, denotes the probability that an instance x with a clean label $Y = i$ will transition to a noisy label $\tilde{Y} = j$, under the condition $Y = i$ and $X = x$. Existing methods have been able to learn statistically consistent classifiers when $T(x)$ is provided [4, 7-11]. By leveraging the transition matrix and the posterior probability of the noisy-class $P(\tilde{Y}|X)$, the clean-class posterior $P(Y|X)$ can be inferred [12].

$$T(x)[P(Y = 1|x),\ldots, P(Y = C|x)]^T = \left[P(\tilde{Y} = 1|x),\ldots, P(\tilde{Y} = C|x)\right]^T \ldots \quad (1)$$

In other words, the clean-class posterior can be inferred by applying $T(x)$ to the noisy-class posterior. However, in most cases, the transition matrices are not given and need to be estimated. Without any other assumptions, obtaining the transition matrix for a specific instance requires the availability of its clean-label information.

3.2. The overview of the network

In the context of learning with instance-dependent label-noise, it is necessary to estimate the noisy transition matrix for each input instance, and it is crucial to carefully consider the presence of ambiguous patterns within it. However, the $T(x)$ in the real world is always hard to know and there should be a efficient way to learn the noisy transition matrix in machine learning. Traditional methods, however, struggle to parametrically learn the clean label transition matrix owing to clean labels that are not readily accessible. Consequently, this paper considers utilizing a subset of clean labels, which are provided in advance, to estimate the $T(x)$ for the entire dataset. Based on the ideas we proposed earlier, we constructed a label transition network to train the dataset (Fig 1). Based on this network, we introduced a specialized loss function, which allows us to learn the $T(x)$ and a good classifier after training.

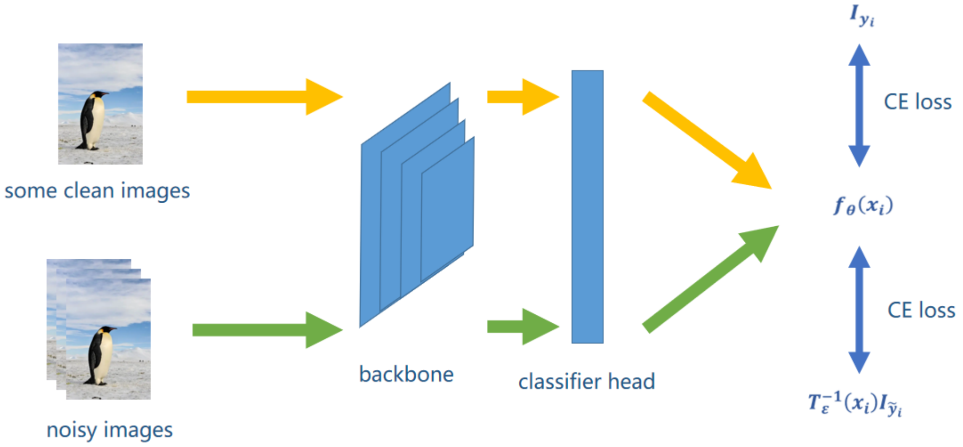

Figure 1. The flow chart of label transition network.

3.3. The loss function

The overview of the network we proposed are showed in the Fig 1. In this network, firstly, both the clean data and noisy data pass through a backbone, which extracts their representative features. After the extraction is completed, then they passe through a classifier head for classification. In the end, we will obtaining a predicted value $\hat{y}$, and we will use a new loss function, proposed in this paper, to pass the loss backward for gradient descent.

3.3.1. The first item in the loss function. The process indicated by the yellow arrow in the Fig 1 pertains to training and predicting on a clean dataset. After going through the above process, we minimize the empirical risk by comparing the predicted values with the clean labels from the clean dataset.

$$\lambda_1 \frac{1}{m} \sum_{i'=1}^{m} l\left(f_\theta(x_{i'}), I_{y_i'}\right) \tag{2}$$

Where x_i is the features of a clean image, λ_1 is weight coefficient, which are usually depend on the size of clean data and the model performance. m is the size of the clean data. l is a known loss function, cross entropy function. f_θ represents the classifier header with the unknown parameters θ that need to be learned. $I_{y_i'}$ represents the clean labels obtained from the clean dataset.

3.3.2. The second item in the loss function

The process indicated by the green arrow in the Fig 1 pertains to training and predicting on the entire dataset. This process is similar to the one for the clean dataset, but in this item, we aim at the noisy data and the noisy transition matrix. We minimize the empirical risk by comparing the predicted values with another transformed noisy data.

$$\frac{1}{n} \sum_{i=1}^{n} l\left(f_\theta(x_i), T_\varepsilon^{-1}(x_i) I_{\tilde{y}_i}\right) \tag{3}$$

Where x_i is the features of a noisy image, n is the size of the noisy data,$T(x)$ represents the noisy transition matrix with an unknown parameter ε. $I_{\tilde{y}_i}$ represents the noise labels obtained from the entire noisy dataset.

3.3.3. The whole loss function

Based on the two empirical risk minimizations mentioned in Section 3.2.1 and 3.2.2, we propose this specialized loss function.

$$L = \frac{1}{n} \sum_{i=1}^{n} l\left(f_\theta(x_i), T_\varepsilon^{-1}(x_i) I_{\tilde{y}_i}\right) + \lambda_1 \frac{1}{m} \sum_{i'=1}^{m} l\left(f_\theta(x_{i'}), I_{y_i'}\right) \tag{4}$$

Where the parameters are discussed in section 3.3.1 and 3.3.2

After obtaining the two items of this loss functions, in order to ensure the accuracy and consistency of the classifier head f_θ's predictions, we mix these two items altogether and minimize the empirical risk. By doing so, upon completion of training, we are able to obtain the accurate noisy transition matrix *T(x)* and a classifier f with excellent performance.

4. Experiment

The experiment setup includes the datasets used, implementation details, and contrast methods in Section 5.1. We will then present and analyze the experimental results on synthetic and real-world noisy datasets to demonstrate the effectiveness of our proposed method in Section 5.2. Additional details on the noise generation algorithm, comparison results and ablation studies can be found in the Appendix experiment setup. We introduce the dataset we used to evaluate the proposed method, and we split it to two parts: noisy data part and clean data part before training our model.

4.1. Dataset

we use the Fashion-MNIST dataset to verify the effectiveness of our method. Fashion-MNIST is a collection of Zalando's product images that includes 60,000 examples in the training set and 10,000 examples in the test set. Each example is a grayscale image with a size of 28x28 pixels, and is labeled with one of 10 possible classes. Each category has a balanced number of images. Therefore, there won't be an issue of the model having a preference for any particular category during neural network training. For the training set, we choice 30 percent of them to add the gaussian noise, thus that we split the dataset into noisy data and clean data, and those gaussian noise can help us to simulate the data in real-world environment.

4.2. Implement details

For the dataset part, 30% of data is randomly selected in the test set and Gaussian noise is added to it. This step is taken to simulate real-world noisy data. We also added masks to the noisy and clean data to distinguish them during model training. We design a label transition network for Fashion-MNIST. For the sake of brevity, the architecture of label transition network is similar to ResNet-18 to a large degree [13], whereas the difference is the last linear layer modified in a shape as same as noisy transition matrix. This linear layer aims to represent the noisy transition matrix so that we can acquire the relatively accurate noisy transition matrix through checking for the model information after finishing the training. We first use SGD with momentum 0.9 to reduce oscillation in gradient descent and make the model move more smoothly in the direction of the gradient; the batch size is 64 and the initial learning rate of 1e-3 to warm up the network for 5 epochs on the mixture of noisy data and clean data. After that we use cosine annealing technique for the rest 45 epochs. This technique can smoothly reduce the learning rate during the training process to improve the model's training performance and generalization ability. Moreover, Cosine Annealing can also diminish training time and computational resources. It also has better robustness and stability compared with other learning rate scheduling algorithms. What is more, we leverage early stopping technique in our training as well. Early Stopping is a regularization technique used to prevent neural networks from overfitting. The basic idea of Early Stopping is to stop training when the performance on a training set no longer improves, in order to prevent overfitting. The rate of noisy data in the whole dataset is 0.3.

4.3. Contrast

To prove effectiveness of our method, we choice ordinary Resnet18 as our contrast model. To guarantee the fairness, the details are all same as our network including the set of hyper-parameters, the only differences are the Resnet18 does not have the last linear layer, which is used to represent the noisy transition matrix, and the loss function for Resnet18 is just ordinary cross entropy loss between the labels and the predicted result.

4.4. Result

We trained both of the two model 10 times, and we choice the best accuracy as our experiment result. For our method, we finally got 93.56% in test set and we got 88.52% on the Resnet18. The result proves that our method is useful and workable.

5. Conclusion

The noisy transition matrix in the real-world is usually unknown, and we purpose a method to estimate it. Using the neural network with the linear layer to represent the noisy transition matrix and the loss function we proposed, can help learn the accurate noisy transition matrix and a neural network, and it is proved workable and achieve a good performance.

References

[1] Yao Q, Yang H, Han B, Niu G, Kwok J. Searching to exploit memorization effect in learning with noisy labels. In: Proceedings of the 37 th International Conference on Machine Learning. Online. 10789-98. https://doi.org/10.48550/arXiv.1911.02377.

[2] Han B, Niu G, Yu X, Yao Q, Xu M, Tsang I, Sugiyama M. SIGUA: Forgetting may make learning with noisy labels more robust. In: Proceedings of the 37 th International Conference on Machine Learning. Online. 4006-16. https://doi.org/10.48550/arXiv.1809.11008.

[3] Han B, Yao J, Niu G, Zhou M, Tsang I, Zhang Y, Sugiyama M. Masking: A new perspective of noisy supervision. In: Proceedings of 32nd Conference on Neural Information Processing Systems. Montréal, Canada. 5836–46. https://doi.org/10.48550/arXiv.1805.08193.

[4] Liu T, Tao D. Classification with noisy labels by importance reweighting. IEEE Trans Pattern Anal Mach Intell 2016;38(3)447–61. https://doi.org/10.1109/TPAMI.2015.2456899.

[5] Zhang Z, Sabuncu MR. Generalized cross entropy loss for training deep neural networks with noisy labels. In: Proceedings of 32nd Conference on Neural Information Processing Systems. Montréal, Canada. 8792–802. https://doi.org/10.48550/arXiv.1805.07836.

[6] Tanaka D, Ikami D, Yamasaki T, Aizawa K. Joint optimization framework for learning with noisy labels. In: Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT. 5552–60. https://doi.org/10.48550/arXiv.1803.11364.

[7] Goldberger J, Ben-Reuven E. Training deep neural-networks using a noise adaptation layer. In: Proceedings of THE 5th International Conference on Learning Representations. Toulon, France. 1–9.

[8] Yu X, Liu T, Gong M, Tao D. Learning with biased complementary labels. In: Computer Vision – ECCV 2018. Cham. 69-85. https://doi.org/10.1007/978-3-030-01246-5_5.

[9] Xia X, Liu T, Wang N, Han B, Gong C, Niu G, Sugiyama M. Are anchor points really indispensable in label-noise learning? In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. Vancouver, Canada. 6838–49. https://doi.org/10.48550/arXiv.1906.00189.

[10] Xia X, Liu T, Han B, Wang N, Gong M, Liu H, Niu G, Tao D, Sugiyama M. Part-dependent label noise: Towards instance-dependent label noise. In: Proceedings of the 34th Conference on Neural Information Processing Systems. Vancouver, Canada. 7597–610. https://doi.org/10.48550/arXiv.2006.07836.

[11] Li X, Liu T, Han B, Niu G, Sugiyama M. Provably end-to-end labelnoise learning without anchor points. In: Proceedings of the 38 th International Conference on Machine Learning. Online. 1–15. https://doi.org/10.48550/arXiv.2102.02400.

[12] Patrini G, Rozza A, Menon A, Nock R, Qu L. Making deep neural networks robust to label noise: A loss correction approach. In: Proceedings of 2017 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Honolulu, HI. 1944–52. https://doi.org/10.1109/CVPR.2017.240.

[13] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of 2016 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Vegas, NV. 770–8. https://doi.org/10.1109/CVPR.2016.90.