Journal of Korea Multimedia Society Vol. 23, No. 11, November 2020(pp. 1349-1360)
https://doi.org/10.9717/kmms.2020.23.11.1349

DeepCleanNet: Training Deep Convolutional Neural Network with Extremely Noisy Labels

Bekhzod Olimov[†], Jeonghong Kim[††]

ABSTRACT

In recent years, Convolutional Neural Networks (CNNs) have been successfully implemented in different tasks of computer vision. Since CNN models are the representatives of supervised learning algorithms, they demand large amount of data in order to train the classifiers. Thus, obtaining data with correct labels is imperative to attain the state-of-the-art performance of the CNN models. However, labelling datasets is quite tedious and expensive process, therefore real-life datasets often exhibit incorrect labels. Although the issue of poorly labelled datasets has been studied before, we have noticed that the methods are very complex and hard to reproduce. Therefore, in this research work, we propose Deep CleanNet - a considerably simple system that achieves competitive results when compared to the existing methods. We use K-means clustering algorithm for selecting data with correct labels and train the new dataset using a deep CNN model. The technique achieves competitive results in both training and validation stages. We conducted experiments using MNIST database of handwritten digits with 50% corrupted labels and achieved up to 10 and 20% increase in training and validation sets accuracy scores, respectively.

Key words: Image Classification, Noisy Labels, Corrupted Labels, CNN.

1. INTRODUCTION

Due to emergence of Deep Learning and development of its models, there was great progress in the computer vision tasks. Especially, image classification and recognition fields have witnessed a considerable improvement by achieving state-of-the-art results using Deep Neural Network (DNN)s. One of the members of DNNs family, Convolutional Neural Network (CNN)s, have been successfully employed to recognize and classify images for the last decade. However, it is undeniable fact that deep networks demand such gigantic amount of data that datasets with thousands or even a couple of million samples are not enough to utilize full power of CNNs. The most well-known models [1-3] that achieved incredible results using CNNs were trained on very large dataset [4] containing more than ten million hand-annotated images. Therefore, even a decade ago, researchers and practitioners were unable to obtain desirable results because of limitations in data amount, but recent rapid increase in image data on the web allowed to acquire required amount of data and to train more powerful and resilient models.

The process of hand-labelling, where human

※ Corresponding Author : Jeonghong Kim, Address: (41566) Daehag-ro 80, Buk-gu, Daegu, Korea, TEL : +82-53-950-5551, FAX : +82-53-950-5551, E-mail : jhk@knu.ac.kr

Receipt date : Sep. 11, 2020, Revision date : Oct. 5, 2020
Approval date : Oct. 19, 2020

[†] Dept. of Computer Science & Engineering, Graduate School, Kyungpook National University
(E-mail : bekhzod.olimov@knu.ac.kr)

[††] Dept. of Computer Science & Engineering, Graduate School, Kyungpook National University

※ This study was supported by the BK21 Plus Project (SW Human Resource Development Program for Supporting Smart Life) funded by the Ministry of Education, School of Computer Science and Engineering, Kyungpook National University, Korea (21A20131600005).

experts are involved to label raw data is exceedingly costly. Additionally, raw images might look so confusing and complicated that even experts in the area may have different opinions regarding the images label. Therefore, the other methods of labelling with no human experts intervention are preferable in majority of the cases. Crowdsourcing and online queries are clear examples of them, where data is labelled by non-expert individuals. Due to insufficient knowledge in the field and subjective judgment of the people, datasets often have great number of incorrect labels, or even worse they can be corrupted. As a result, it negatively impacts on the performance of a deep CNN model, because it learns from wrong labels and accepts them as correct, consequently fails to properly classify input data. This fact was also proven in [5], where the authors found out that CNNs can memorize even very large datasets. Also, they noticed that in the case of corrupted labels existence, a model failed to generalize well to unseen data with correct labels because it learned from incorrectly labelled data.

In general, corrupted labels in dataset might cause large problems in all types of supervised learning algorithms. Nevertheless, it is common to face poorly labelled datasets in the wild due to the aforementioned problems. As the goal of image classification problem is to classify images with high accuracy, we need to find the methods to address the problem of corrupted labels. The first and most simple thought is just to select correctly labelled images manually. But this process is exceedingly tedious and tiresome. Moreover, when the dataset contains millions of samples, manual selection of correctly labelled data might become enormously time-consuming.

Since the issue of poorly labelled data is essential in training CNNs, training a model with corrupted labels has been broadly researched so far. We can roughly divide existing techniques into two subgroups, namely statistical methods as well as deep learning methods, which concentrate on training deep neural networks with poorly labelled data. They will be discussed in more details in the second section of the work. However in general we can point out that majority of those techniques have a number of shortcomings. Firstly, majority of the methods heavily depend on very complex algorithms. Second, due to complexity of the methods, it is very difficult to reproduce them and use for other datasets. Third, most of them trained two deep neural networks simultaneously, which is both time and computationally expensive. In general, the proposed method addresses aforesaid aspects and contributes for the improvement of the area in following ways:

- The technique is relatively simple in comparison to the existing methods and contains fewer complex steps.
- It can be effortlessly comprehendible and easily reproducible.
- The proposed approach is efficient in terms of time and computation.
- It deals with the 50% incorrectly labelled dataset.
- The results of the method are competitive with the existing state-of-the-art techniques.

The rest of the research work is organized as follows. Section 2 contains the brief introduction of the previous research related to our method. The methodology of our method is described in Section 3. The information about the experiments and their results are provided Section 4. Finally, Section 5 summarizes this work with conclusion and directions for the future work.

2. RELATED WORKS

2.1 Conventional algorithms to train models with noisy labels

Great amount of research has been conducted on the detrimental outcomes of training a model with the data containing incorrect labels that produced various solutions to tackle the issue. As mentioned above, existing techniques might be categorized into statistical and deep learning methods. Regarding the former, they mainly contributed to the tackling corrupted labelled data problem theoretically [6]. To illustrate, Natarajan et. al studied the problem binary classification with the existence of random noise and proposed a simple unbiased estimator as well as weighted surrogate loss [7]. Menon et. al used class probability estimation to study noisy labels and identified that balanced error can be optimized with no knowledge of corrupted labels and range of classification risks can be minimized [8]. Liu et al. also studied a classification problem with corrupted labels [9] and demonstrated that surrogate loss, while used with importance weighing, can be successfully used for classification task with noisy labels on both synthetic and real datasets. Bootkrajang et. al proposed new regularization method [10] that deals with noise in high dimensions and demonstrated its usage in concrete applications.

2.2 Deep learning approaches to train models with noisy labels

This subgroup of the methods to deal with corrupted labels consists of solutions for Deep Learning models. For example, Bekker et. al proposed an approach [11] that ignores the presence of incorrect labels learns from neural network parameters as well as the noise distribution at the same time. Mnih et. al proposed different loss functions [12] that can tackle wrongly labelled data issue and train deep neural networks on complex datasets. Sukhbaatar et. al presented a method [13], which matches output of the model with the corrupted label distribution. A new crowd layer that enables to train end-to-end deep neural networks using corrupted labels was introduced by Rodrigues et. al [14]. Tanaka et. al presented a combined optimization framework [15] that can fix incorrect labels during training stages along with model's other parameters. Veit et. al [16] showed the efficiency of firstly train with noisy data and then fine-tune with clean data. The method resulted in impressive results in very large dataset with almost 10 million samples.

2.3 The-state-of-the-art approaches to train models with noisy labels

Apart from the works mentioned above, perhaps the most influential techniques on dealing with corrupted labels were S-model [17], Bootstrap [18], F-correction [19], Decoupling [20], and Co-teaching [6]. The first several methods greatly contributed to the progress of corrupted labels problem solution, whilst the latter improved Decoupling method by addressing its shortcomings.

The authors of S-model proposed a technique that can be trained using only noisy labelled data and showed that learning is possible without any clean data. They also demonstrated that addition of softmax output layer allows to employ the algorithm even with deep neural networks. Bootstrap technique creators presented an algorithm that creates noise distribution matrix that maps predictions of the model to the targets. The loss computed from the mapping allows the model to explore the noisy data characteristics. Similarly, F-correction method also relies on the noise transition matrix building. Its essence is the correction of loss by the noise transition matrix. In the first stage, a regular model is trained to build the noise transition matrix and then another model is used to make predictions based on the earlier defined matrix.

The essence of decoupling technique is to let the classifier decide on its own whether to update the model or not by handling each sample of the dataset one by one. Also, the classifier's ability to perform huge number of updates in the beginning of the training and slowly decrease the updates in the end remained intact. In order to achieve this, the authors trained two deep models and updated them only when the predictions were not the same. However, Decoupling technique could not deal with noisy labels in a detailed way. Co-teaching technique trained two networks too, but the method's novelty was to permit to filter various kinds of errors caused by noisy labels. By allowing this, the authors improved Decoupling approaches, which slowly accumulated the error because the error from the first classifier was sent back to itself in every following mini-batch data. Co-teaching method showed its power even in exceedingly corrupted data with the noise rate of 50%.

3. THE PROPOSED METHODOLOGY

As we mentioned earlier, we use an unsupervised learning algorithm followed by a deep CNN model. The graphical illustration of the method is depicted in Fig. 1.

The proposed approach for dealing with noisy labels comprised of following stages:

- In the first step, the dataset with corrupted labels is divided into the several parts. The number of divided parts is determined by the number of classes in the dataset.
- Once the dataset is divided into several slices, each of it goes through K-means clustering unsupervised learning algorithm, which separates the input into two clusters.
- After visual inspection of the clusters, the one with majority of correct labels is selected.
- n number of clusters then are concatenated in order to create a new dataset that contains training examples with relatively correct labels.
- The new dataset is trained using deep CNN that is shown in Fig. 2.
- The outcome of the method is more smooth and better training process with reduced loss as well as improved accuracy.

3.2 K-means clustering algorithm

As stated in the description of the proposed method, we utilize K-means clustering algorithm to obtain data samples with correct labels. K-mean clustering algorith [21], [22] finds a fixed number of clusters, initially defined by a user, in the dataset. A cluster is a collection of data samples, which are collected together based on similarities in the features. In order to make the algorithm work, first we need to define "k", the number of clusters we desire. Another important term, centroid, defines a cluster centers, so after setting "k", the algorithm finds cluster centroids that are the locations of the center of each cluster. In general, K-means finds "k" clusters and assigns all data samples to the nearest cluster by targeting to keep the clusters as small as possible. The objective of the algorithm is to minimize the sum of squared

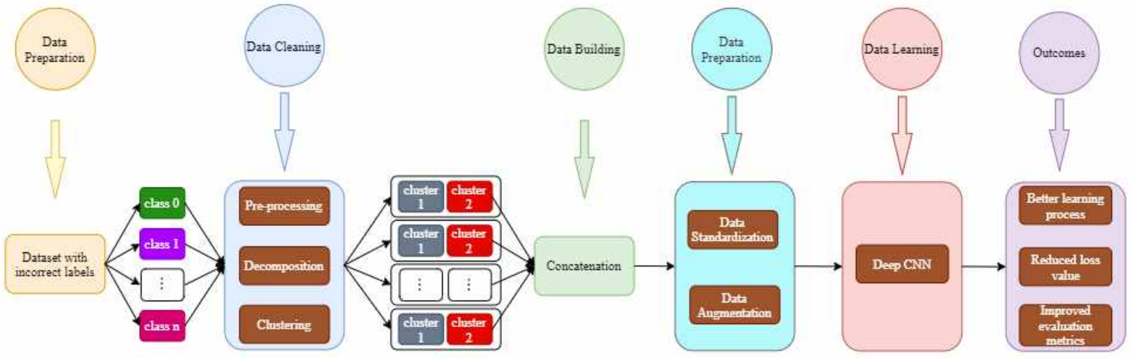


Fig. 1. The methodology of the proposed scheme.

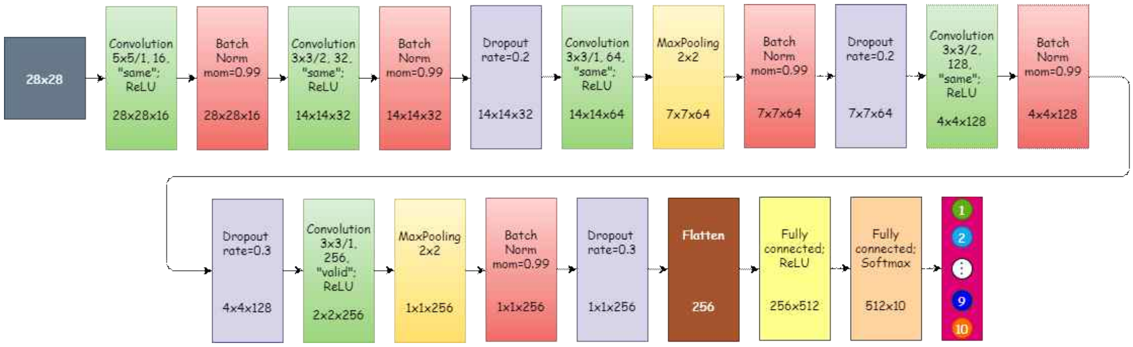


Fig. 2. Deep CNN model architechture for the experiements.

distance between the data points and their corresponding cluster centers:

$$\underset{X}{\arg\min} \sum_{l=1}^{K} \sum_{x_i \in X_l} \left\| x_i - \mu^l \right\|^2 \tag{1}$$

In order to attain the objective function, the algorithm randomly selects the coordinates, which are used as the first cluster center locations. Afterwards, K-means algorithm conducts following steps:

* Assigns each sample in the dataset to the nearest cluster based on Euclidean distance:

$$d(a,b) = \sqrt{\sum_{i}^{n} (a_i - b_i)^2} \tag{2}$$

* For every centroid, computes mean values of every member of the cluster and sets the mean value as the new value of the cluster centroid.

The algorithm stops processing in following cases:

* Objective function is completely optimized - the next iteration of the algorithm cannot improve the optimal location of centroids.

* Maximum number of iterations set by a user are completed.

3.3 Deep CNN Architecture

Regarding model architecture, we decided to use a CNN model for our experiments. Because CNN has been consistently achieving state-of-the-art results in image classification tasks. The architecture of our model is illustrated in Fig. 2.

Fig. 2 shows general structure of the CNN model. The first hidden layer contains convolutional operation with 16 kernels each size of 5×5, stride of 1, which is responsible for the step of kernels across the image and “same” that keeps the image size intact. The output of the convolutional layer then passes through ReLU activation function followed by a batch normalization layer, which assists to keep the activations normalized. The second hidden layer has similar structure, first 32 kernels that have 3×3 size do convolution operation with the stride of 2 and “same” padding. The output then goes through ReLU activation function and batch normalization layers. The only difference with the first hidden layer is the addition of Dropout layer, which is responsible for diminishing overfitting problem. Since the second hidden layer is only the beginning of the training, we believe that the model will not overfit to training samples yet, so we chose a negligible 0.2 rate for dropout, which randomly drops 20% of the nodes in the hidden layer 2. The third hidden layer’s convolutional operation is done with 64 kernels that are 3×3, stride of 1 and “same” padding followed by ReLU activation function. Here, we use max pooling layer to decrease the size of the image by two. The output of it passes regular batch normalization and dropout layers. The convolution operation of the hidden layer 4 is done with 128 3×3 filters with the stride of 2 and “same” padding. Its output then goes through ReLU activation function, batch nor-

Table 1. General information about MNIST database of handwritten digits with noisy labels

Class labels	Training examples	Validation examples
0	5963	1000
1	6332	1000
2	5678	1000
3	6061	1000
4	5932	1000
5	5741	1000
6	5968	1000
7	6115	1000
8	5931	1000
9	5979	1000

malization and dropout operations. For the last convolution operation, we use 256 filters with 3x3 kernel size, stride of 1 and “same” padding. Then, we apply activation function, max pooling, batch normalization and dropout operations before flattening the image in order to create an input for fully connected layer. The output of it passes through another fully connected layer using softmax activation function, which then outputs 10 values for each of the classes.

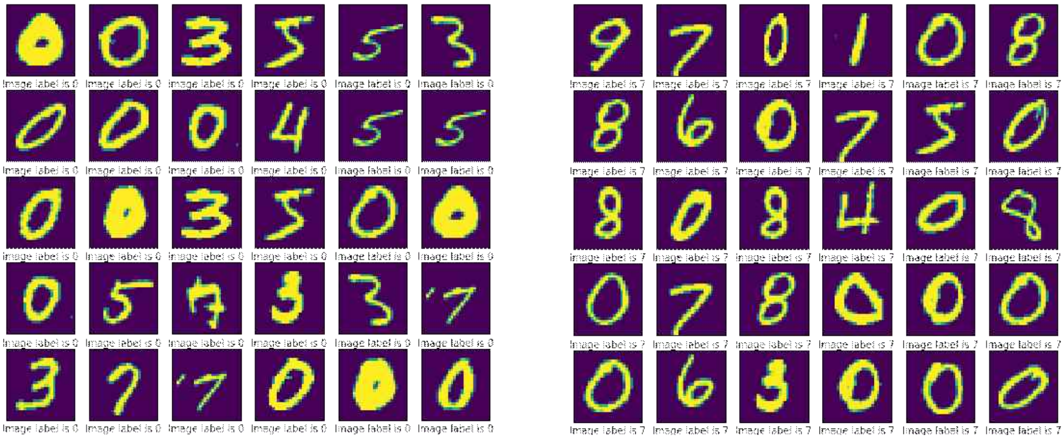
4. EXPERIMENTS AND RESULTS

4.1 Dataset

We use MNIST handwritten digits database, which has 50% incorrect labels for the samples. It is considered as extremely noisy dataset where each second training example has been wrongly labelled. The dataset contains sixty thousand training samples for training and ten thousand examples for validation accordingly. It has same distribution of training examples like in original MNIST database of handwritten digits [23] that comprises various number of examples in each class. General description of the dataset for the experiments is given in Table 1.

As can be seen from Table 1, number of training examples are different in contrast to validation examples, which have equal one thousand examples for each class. In order to make sure the dataset training examples contain corrupted labels, we illustrated them in Fig. 3.

Fig. 3 depicts randomly chosen training examples from the classes of 0 and 7. It is obvious that dataset is labelled incorrectly, since in (a) there are only 13 images, which actually look like 0, whilst the rest ones are totally different digits. The same tendency can be observed in (b), which has only



(a) (b)

Fig. 3. Training examples of the corrupted data. (a) training images labelled as 0 (b) training images labelled as 7.

a few images of number 7.

4.2 Training Setup

We formualted a deep CNN model using Keras library with Python programming language and conducted experiments using NVIDIA GeForce RTX 2060 SUPER GPU. For all experiments, we initialized weights using Kaiming weight initialization [3] and used Adam optimizer [24] with momentum of 0.9 and learning rate of 1e-3. Additionally, loss function was sparse categorical cross-entropy and evaluation metrics was accuracy as in [25]. We also trained the classifier for 50 epochs with the batch size of 256.

The overall picture of our experiments is depicted in Fig. 4. We began testing the model by separating the images of each class into different folders. Since there are ten classes in the dataset, we came up with ten folders each of which contain corrupted labels.

In the second stage of the proposed method, we use unsupervised learning algorithm to clean the dataset containing corrupted labels. Since this sort of learning does not require any labels, in contrast it creates labels based on the similar features of data points. We assumed that even though labels were incorrect, unsupervised learning can find similar samples based on their features and create clusters of training examples with alike parameters.

For our experiments we used one of the most efficient representatives of unsupervised learning family - K-means clustering algorithm. We set the number of clusters as 2, because we expected the algorithm to group the samples into two clusters, like the digits that correspond to the class label (correct labelled data) and the digits that are different from cluster 1 that does not look like to the digits with corresponding class label. In other words, in case of digit 0, we anticipated to group all 0 to one cluster, and all other digits (since their features are different) to another cluster. This logic worked quite well in our experiments. The results of K-means clustering algorithm usage can be seen in Fig. 5.

As can be seen from Fig. 5, K-means clustering algorithm greatly assisted to extract somehow similar images from mixed training examples. Fig. 5 (a) and (b) as well as (c) and (d) demonstrate the division of the training images from Fig. 1 (a) and (b) respectively. Originally, they should be la-

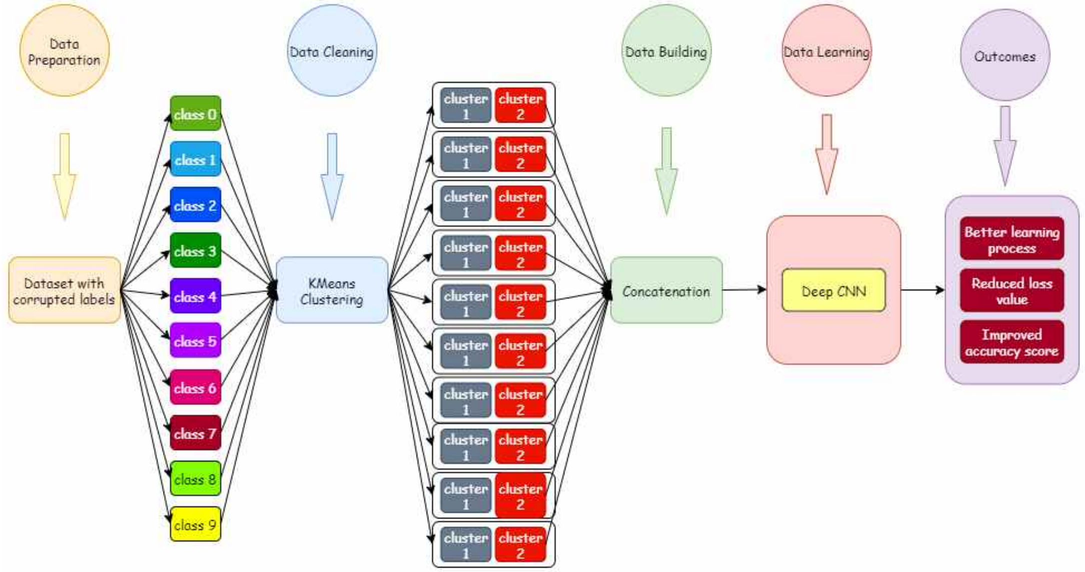
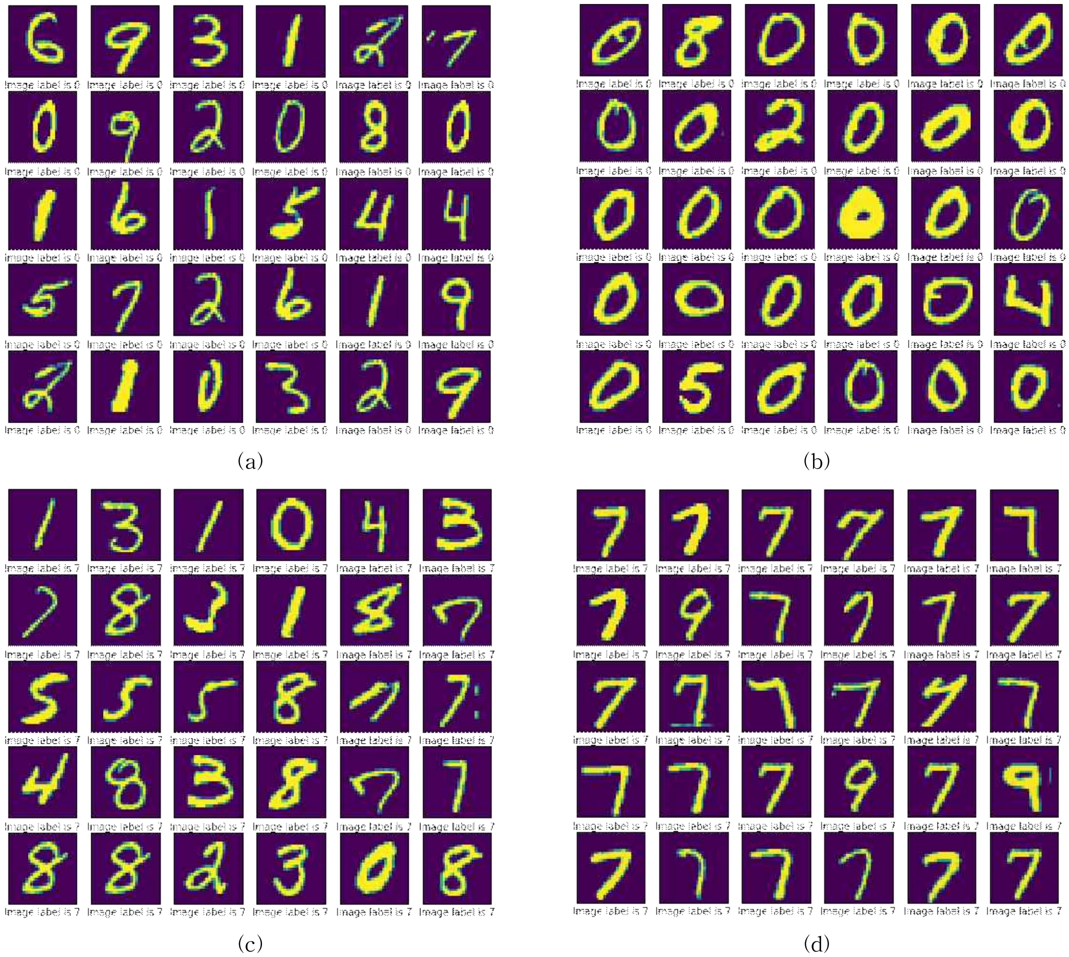


Fig. 4. The graphical illustration of the experiment using MNIST digits database with corrupted labels.



(a)

(b)

(c)

(d)

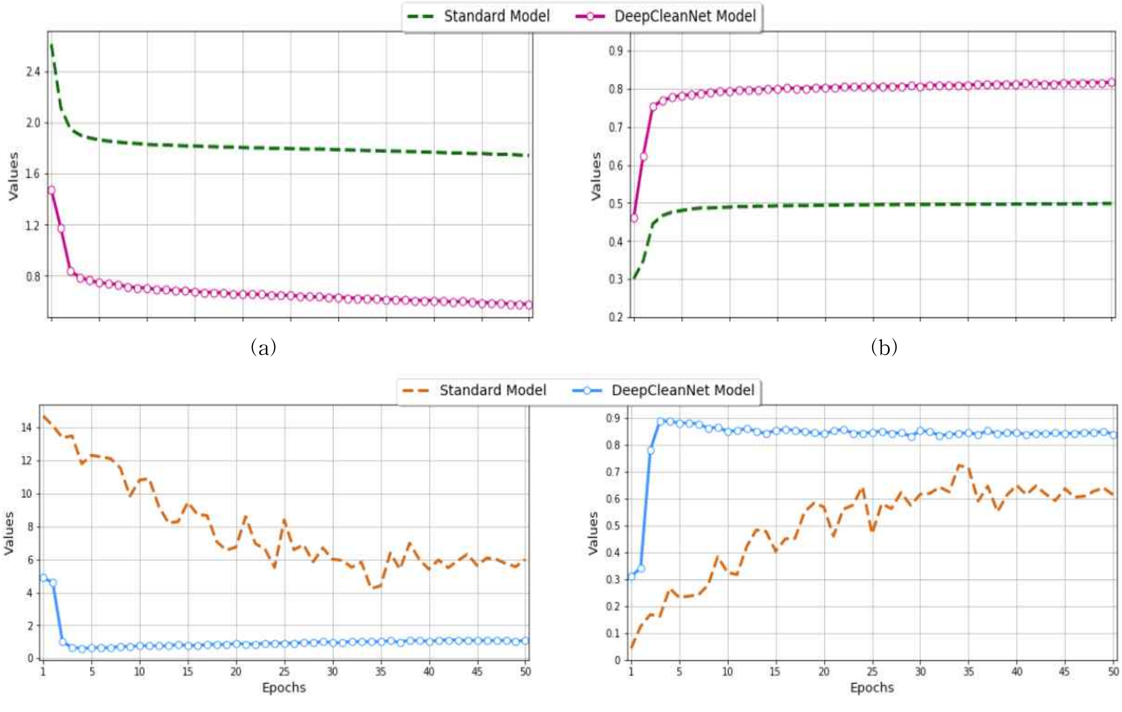
Fig. 5. Training examples of the corrupted data after applying K-means clustering algorithm. (a) cluster 1 of images labelled as 0 (b) cluster 2 of images labelled as 0 (c) cluster 1 of images labelled as 7 (d) cluster 2 of images labelled as 7.

belled as 0 and 7, since they are the classes for 0 and 7 digits accordingly. Admittedly, K-means clustering is not impeccable and cannot separate the training examples flawlessly. However, the output of the algorithm made great progress in in some way cleaning the training images from corrupted labels. Additionally, it should be noted that the process of making two distinct clusters takes only a few seconds so for 10 classes we spent approximately half a minute, which is thousands of times faster than tedious manual cleaning. After obtaining the new dataset with by some means correct labels for each class, we concatenated the training images into one in order to create an appropriate input for upcoming step - classification using CNN model.

4.3 Experimental Results

In this subsection, we compared the experimental results of the baseline and proposed methods inspired by [26]. In order to compare the considered models, we first trained a CNN model using totally corrupted data, then repeated the process by employing cleaned data. The outcome of the experiments can be seen in Fig. 6.

We can observe the difference between the re-



(a)

(b)

(c)

(d)

Fig. 6. Results of the experiment. (a) training loss (b) training accuracy (c) validation loss (d) validation accuracy.

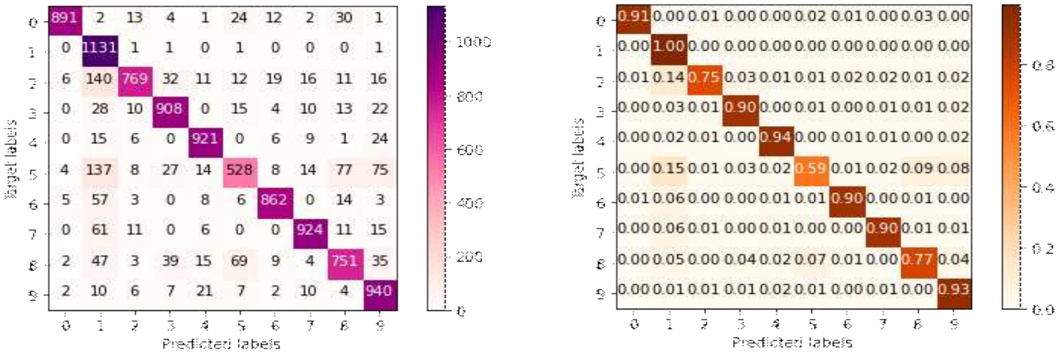
sults of training process with corrupted labels and the proposed method in Fig. 6. It is noticeable that both training loss and training accuracy increase, while validation loss and accuracy decrease respectively in the case of training with incorrect labels, because the model accepts wrongly labelled images as correct and make predictions based on this knowledge. Consequently, validation data that contains correct labelled data suffers from incorrect learning and the model experiences downward tendency in validation loss and accuracy, which we care most. In contrast, regarding the case of the proposed method, training, validation loss and accuracy are comparably stable, and the loss steadily decreases while the accuracy gradually increases as we expect.

Moreover, it should be noted that the proposed technique performed significantly better than the training with corrupted labels in the aforesaid aspects. Specifically, in the case of the standard model the training loss and validation loss converged at approximately 1.8 and 6.0 that is significantly higher than the results of the proposed model, which obtained 0.5 and 1.12 units for training and validation loss, respectively. Similarly, the training and validation accuracy of the proposed model were both nearly 80%, while the standard model obtained 30% and 20% less accuracy in training and validation sets in comparison with the proposed model by converging at 50% and 60% accuracy, accordingly.

5. DISCUSSION

After successfully training the model and obtaining satisfactory results, we decided to plot confusion matrices in order to observe the performance of the network in classification of each label.

As can be seen from Fig. 7, the model trained with the proposed method performed quite well in



(a)

	0	1	2	3	4	5	6	7	8	9
0	891	2	13	4	1	24	12	2	30	1
1	0	1131	1	1	0	1	0	0	0	1
2	6	140	769	32	11	12	19	16	11	16
3	0	28	10	908	0	15	4	10	13	22
4	0	15	6	0	921	0	6	9	1	24
5	4	137	8	27	14	528	8	14	77	75
6	5	57	3	0	8	6	862	0	14	3
7	0	61	11	0	6	0	0	924	11	15
8	2	47	3	39	15	69	9	4	751	35
9	2	10	6	7	21	7	2	10	4	940

(b)

	0	1	2	3	4	5	6	7	8	9
0	0.91	0.00	0.01	0.00	0.00	0.02	0.01	0.00	0.03	0.00
1	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.01	0.14	0.75	0.03	0.01	0.01	0.02	0.02	0.01	0.02
3	0.00	0.03	0.01	0.90	0.00	0.01	0.00	0.01	0.01	0.02
4	0.00	0.02	0.01	0.00	0.94	0.00	0.01	0.01	0.00	0.02
5	0.00	0.15	0.01	0.03	0.02	0.59	0.01	0.02	0.09	0.08
6	0.01	0.06	0.00	0.00	0.01	0.01	0.90	0.00	0.01	0.00
7	0.00	0.06	0.01	0.00	0.01	0.00	0.00	0.90	0.01	0.01
8	0.00	0.05	0.00	0.04	0.02	0.07	0.01	0.00	0.77	0.04
9	0.00	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.00	0.93

Fig. 7. Confusion matrices. (a) Confusion matrix without normalization. (b) Confusion matrix with normalization.

classifying the handwritten digits with 50% incorrect labels. In 7 out of 10 classes, the model attained accuracy of 90% and higher, with an impeccable result for digit 1. Class 2 and 8 had similar classification accuracy rates of approximately 75%, whilst class 5 had pretty low score of about 60%.

In fact, the network could not classify the handwritten digits perfectly and misclassified several categories. Specifically, the model mostly misclassified digit 0 as 8 (30 times) and 5 (24 times), while it could recognize digit 1 almost perfectly with only few misclassifications, such as 2,3,5, and 9. Regarding digit 2, it primarily mispredicted as 1 and digit 5 was misclassified as 1 in majority cases. The most confused classes by the model however, were categories of 8 and 9 because their mispredicted labels were various, ranging from 0 to 9 excluding 8 fro class 8 and ranging from 0 to 8 for class 9, respectively.

Such imperfect performance of the model was mainly because it used incorrectly labeled data for training. However, considering significant increase in its performance in comparison with the standard model, we can conclude that the proposed model has promising application perspectives in the domain of training CNN models with corruped labels.

6. CONCLUSION

To summarize, we conducted research into training a model using dataset with incorrect labels. We did extensive literature review on various statistical and deep learning methods to alleviate the consequences of the issue. Based on the knowledge from related work, we realized that existing approaches of training with corrupted labels are either complex or hard to reimplement. Therefore, we decided to propose our method that is simple yet effective. The technique is so easy to use that requires very little engineering skills to be implemented. To show the competitive performance of the technique, we selected MNIST database of handwritten digits with 50% incorrect labels for our experiments. Also, we employed K-means unsupervised learning algorithm to choose data with somehow correct labels and trained the new dataset with deep CNN model. The results were satisfactory. The proposed method with data selection outperformed training with 50% corrupted labels by more than 20% and 10% for training and validation accuracies accordingly. Additionally, our method made the learning process more smooth and stable attaining steady decrease of loss and gradual increase of accuracy for both training and validation sets. The proposed method not only attains competitive outcomes, but also allows the data collection process to be less costly and time-consuming.

Although the proposed method attains desired results of dealing with corrupted labels, it can be further improved in the future. One direction for improvement is to find an approach to keep majority of the data points in the training set. Because, in our case the number of training examples decreased twice due to 50% corrupted labels. We will conduct additional research to find out the ways to efficiently extract correct labelled data with keeping more training datapoints in comparison to the outcome of this method.

REFERENCE

[1] A. Krizhevsky, I. Sutskever, and G.E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems*, Vol. 13, No. 2, pp. 1097-1105, 2012.

[2] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-scale Image Recognition," *Proceeding of 3rd International Conference on Learning Representations, ICLR 2015-Conference Track Proceedings*, pp. 132-137, 2015.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-level Performance on Imagenet Classification," *Proceeding of IEEE International Conference on Computer Vision*, pp. 1026-1034, 2015.

[4] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-scale Hierarchical Image Database," *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248-255, 2009.

[5] C. Zhang, B. Recht, S. Bengio, M. Hardt, and O. Vinyals, "Understanding Deep Learning Requires Rethinking Generalization," *Proceeding of 5th International Conference on Learning Representations, ICLR 2017-Conference Track Proceedings,* pp. 301-307, 2019.

[6] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, et al., "Co-teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels," *Advances in Neural Information Processing Systems,* Vol. 11, No. 3, pp. 8527-8537, 2018

[7] N. Natarajan, I.S. Dhillon, P. Ravikumar, and A. Tewari, "Learning with Noisy Labels," *Advances in Neural Information Processing Systems*, Vol. 13, No. 6, pp. 1196-1204, 2013.

[8] A.K. Menon, B.V. Rooyen, C.S. Ong, and R.C. Williamson, "Learning from Corrupted Binary Labels via Class-probability Estimation," *Proceeding of 32nd International Conference on Machine Learning,* pp. 125-134, 2015.

[9] T. Liu and D. Tao, "Classification with Noisy Labels by Importance Reweighting," *IEEE Transaction Pattern Analysis and Machince Intelligence*, Vol. 13, No. 3, pp. 447-461, 2016.

[10] J. Bootkrajang and A. Kabán, "Label-noise Robust Logistic Regression and its Applications," *Proceeding of Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 143-158, 2012.

[11] A.J. Bekker and J. Goldberger, "Training Deep Neural-networks Based on Unreliable Labels," *Proceeding of IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2682-2686, 2016

[12] V. Mnih and G. Hinton, "Learning to Label Aerial Images from Noisy Data," *Proceedings of the 29th International Conference on Machine Learning,* pp. 567-574, 2012.

[13] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus, "Training Convolutional Networks with Noisy Labels," *Proceeding of 3rd International Conference Learning Representation ICLR 2015-Workshop Track Processing*, pp. 1-11, 2015.

[14] F. Rodrigues and F.C. Pereira, "Deep Learning from Crowds," *Proceeding of 32nd Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence,* pp. 1233-1242, 2018.

[15] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa, "Joint Optimization Framework for Learning with Noisy Labels," *Proceedings of*

the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 5552-5560, 2018.

[16] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie, "Learning from Noisy Large-scale Datasets with Minimal Supervision," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* pp. 839-847, 2017.

[17] J. Goldberger, E. Ben-reuven, and E. Faculty, "Training Deep Neural Networks Using a Noise Adaptation Layer," *Proceeding of International Conference on Learning Representations*, pp. 1-9, 2017.

[18] S.E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training Deep Neural Networks on Noisy Labels with Bootstrapping," *Proceeding of International Conference on Learning Representations, ICLR 2015-Workshop Track Proceedings*, pp. 91-97, 2015.

[19] G. Patrini, A. Rozza, A.K. Menon, R. Nock, and L. Qu, "Making Deep Neural Networks Robust to Label Noise: A Loss Correction Approach," *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition,* pp. 1944-1952, 2017.

[20] E. Malach and S. Shalev-Shwartz, "Decoupling "When to Update" from "How to Update," *Advances in Neural Information Processing Systems,* Vol. 8, No. 4, pp. 960-970, 2017.

[21] D.T. Pham, S.S. Dimov, and C.D. Nguyen, "Selection of K in K-means Clustering," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science,* pp. 103-119, 2005.

[22] T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, and A.Y. Wu, "An Efficient K-means Clustering Algorithms: Analysis and Implementation," *IEEE Trans. Pattern Analysis and Machince Intelligence*, Vol. 3, No. 9, pp. 881-892, 2002.

[23] The MNIST Database of Handwritten Digits (1998), http://yann.lecun.com/exdb/mnist/ (accessed September 28, 2020).

[24] D.P. Kingma and J.L. Ba, "Adam: A Method for Stochastic Optimization," *Proceeding of International Conference on Learning Representations, ICLR 2015-Conference Track Proceedings,* pp. 302-308, 2015.

[25] T. Kankh, B. Baek, S. Kim, L. Do, W. Yoon, I. Park, et al., "Assessment of ASPECTS from CT scans using Deep Learning," *Journal of Korea Multimedia Society,* Vol. 22, No. 5, pp. 573-579, 2019.

[26] Y. Seol, Y. Kim, K. Nam, K. Kim, "Comparison on the Deep Learning Performance of a Field of View Variable Color Images of Uterine Cervix," *Journal of Korea Multimedia Society,* Vol. 23, No. 7, pp. 812-818, 2020.

Bekhzod Olimov

received his B.S. degree in economics from Fergana Polytechnic Institute, Uzbekistan and obtained M.S. degree from Yeungnam University, South Korea in 2014 and 2018, respectively. He is currently enrolled as a PhD student in computer science and engineering department of Kyungpook National University, South Korea.

His research interests are computer vision and pattern recognition using deep learning algorithms.

Jeonghong Kim

received the B.S. and M.S degrees from Kyungpook National University, Daegu, South Korea and the Ph.D. degrees from Chungnam National University, Daejeon, South Korea.

He was a Visiting Professor at Boise University, Boise, USA, from 2013 to 2014. His current research are bio signal processing and pattern recognition using deep learning algorithms.