Assignment 2

Tutors: Songhua Wu
Group members:
1. Tulika Srivastava | 500657898 | tshr3215@uni.sydney.edu.au
2. Pooja Vijay Mahajan | 510282930 | pmah0895@uni.sydney.edu.au

Abstract

Label noise is a major challenge in the era of big data to implement supervised machine learning problems and deep neural networks [1]. It is defined as the presence of incorrectly classified target labels in the dataset occurring due to lack of sufficient discriminatory information, subjectivity of labelling and errors produced by non-expert labellers [2]. The objective of this experiment was to construct a transition matrix estimator using anchor points. This was further used to design classifier using forward learning. Two loss functions, forward learning approach and early-learning regularization were used to analyse the model's robustness to label noise in three real world datasets, namely FashionMINIST0.3, FashionMINIST0.6 and CIFAR. The organization of the report is as follows. In section 1 of the report, there is a brief overview of label noise and the methods used in the experiment. In section 2, the literature review discusses the problem of learning with label noise along with its relevance and applications in various domains. Section 3 consists of a detailed description of the classification models along with the formulation and foundation of cost functions and optimization algorithms. In section 4, the entire experimental setup of datasets, methods, evaluation metrics, has been discussed followed by a rigorous analysis of the results. Section 5 consists of a summary of results and some insights on research gaps for future work.

1 Introduction

1.1 Problem and it's significance

Label noise is defined as a situation in which the original label given to the instance is misclassified [2]. In comparison to the attribute noise, previous studies have shown that label noise is more detrimental compared to attribute noise and lowers the overall performance of the machine learning model [2]. Since it has an impact on various fields and applications in machine learning and deep learning, it is crucial to comprehend the methods that can be used to handle noisy labels [2]. Label noise alters the relationship between the measured output and features to some extent [3]. Researchers have found additional downsides, such as difficulty in performing feature selection, a substantial increase in the complexity of learning models, and the need to increase the training data in order to produce the same test accuracy [3]. In general, label noise can occur due to several reasons [4]. Expert labellers can make mistakes in assigning the right label [4]. Categorization of certain instances may be subjective or very confusing [4]. Data corruption can occur due to encoding problem and typing errors [4]. Since labelling is a tedious task, cheaper alternatives such as crowd sourcing or image annotations obtained directly from the web are used which in turn might produce labels of lower quality [4].

1.2 Types of label noise

Label noise can be divided into three broad categories:

1. Random classification noise (RCN): The noise rate is independent of both the feature space and the original true label with the observed labels being different from the true labels [5]. The probability (row) of the flipping of clean label to noisy label is independent of both the features and the true labels and is a constant as shown in Eq. (1) [5].

$$\rho_\gamma(X) = P(\tilde{Y}|Y, X) = P(\tilde{Y}|Y); \rho_{+1}(X) = \rho_{-1}(X) = \rho. \tag{1}$$

2. Class-Conditional Random Label Noise (CCN): The occurrence of label noise depends solely on the clean label, but it is still independent of the feature space [6]. This might occur if instances of certain classes are more prone to be misclassified [6]. This allows us to define a symmetric label noise, and a transition matrix in which the flip rate (row-subscript y) is constant for the labels belonging to a class y as shown in Eq. (2) [6].

$$\rho_\gamma(X) = P(\tilde{Y}|Y, X) = P(\tilde{Y}|Y); \rho_{+1}(X) = \rho_{+1}, \rho_{-1}(X) = \rho_{-1}. \tag{2}$$

3. Instance-and Label-Dependent Noise (ILN): It occurs when the flip rate depends on the features space as well as on the original label [7] as shown in Eq. (3). This leads to modelling of cases in which label noise is more frequent in classification boundaries or more frequent in low density regions in the feature space [7].

$$\rho_\gamma(X) = P(\tilde{Y}|Y, X). \tag{3}$$

   In equations (1), (2) and (3), X is an observation (feature), Y is the clean but unobservable label, and $\hat{Y}$ is the observed noisy label.

1.3 Ways to deal with label noise

The objective of research in this domain is to find label noise aware algorithms which can help reduce the negative influence of label noise on the overall performance quality. To do so, researchers need to find ways to make machine learning models and pipelines more robust to different types of label noise. As per the previous literature, there are various techniques for handling label noise which are discussed in section 2.

1.4 Applications

The proliferation of large datasets sets a dire need for more robust classification algorithms which have a good performance accuracy to classify the wrong labels. Learning with label noise can be used to construct deep self-learning frameworks which are self-sufficient in training a resilient network without any additional monitoring and without any reliability of the assumptions of the distribution of noisy labels [9].

1.5 An overview of methods used in the experiment

In this assignment, different deep learning models like MLP, LeNet, ResNet were explored and their robustness were compared to data with noisy labels by using different techniques like Forward Learning and Early Learning Regularization. The performance evaluation was done using Top-1 accuracy and loss. Throughout the experimental setup, the hyper-parameters were tuned to improve the model's accuracy. The details are provided in section 4. A transition matrix estimation was done for CIFAR dataset using anchor points. The effectiveness of the transition matrix estimator was validated by estimating the transition matrix for FashionMINIST0.3 and FashionMINIST0.6 datasets and comparing the estimated matrix with the original one given as part of the assignment. After performing extensive experimentation, inferences were made based on the methods and varied setup of hyper-parameters as described in section 4.

2 Related Work

Learning with label noise has gained significant attention of researchers as larger datasets are more prone to label noise. Data with noisy labels can prove to be dexterous if it belongs to domains such as medicine, finance, crowdsourcing, semantic segmentation and many more [4]. After extensive research, it was found that there are two broad categories methods used for learning with label noise [10].

1. Noise model-based methods: These methods require the information about the noise as a prerequisite to eliminate the noisy labels from the data [4]. They include noisy channels, label noise cleansing, dataset pruning, sample choosing, importance reweighting and labeller quality assessment [4].
2. Noise model-free methods: It is based on methods which are inherently more robust to label noise without having a prior information of the noise [11]. These include robust loses, meta-learning, regularizers, and ensemble methods [11].

Some of these methods are described below:

2.1 Noisy Channels

When the data with the noisy labels is passed through the loss function, it passes noisy gradients to the noisy channel which supplies noise-free gradients to the model and as a result the model generates noise-free predictions [4]. The noise is represented in the form of a transition matrix. The empirical risk $\hat{R}_{l,D}(f)$ is as mentioned in Eq. (4).

$$\hat{R}_{l,D}(f) = \frac{1}{N}\sum_{i=1}^{N} l(Q(f_\theta(x_i)), \tilde{y}_i), \tag{4}$$

where $Q(f_\theta(x_i)) = p(\tilde{y}_i|f_\theta(x_i))$ is the association between model predictions to render noisy labels [4].

Advantage: The noisy channel inculcates the peculiarities of the noisy labels. This way, the base learner is provided with clean data during training [4].

Disadvantage: As the number of targets/labels increase in the dataset, there is a subsequent increase in the size of the noise transition matrix making it less scalable [4].

2.2 Label noise cleansing

It is based on detecting instances which are suspected of having label noise [12]. Once this instance is detected, we can filter, reweight or relabel based on the label, the model thinks is right [12]. The detection of these instances can be done using scoring methods which are also used in anomaly detection or outlier detection [12]. These instances can be filtered out if the class instance does not agree with the majority class of instances nearer to the feature's space [12]. Other approaches include iterative label cleaning in parallel with training, label smoothing and label cleaning with generative models [12]. The empirical risk $\hat{R}_{l,D}(f)$ is shown in Eq. (5).

$$\hat{R}_{l,D}(f) = \frac{1}{N}\sum_{i=1}^{N} l(f_\theta(x_i), G(\tilde{y}_i, x_i)), \tag{5}$$

where $G(\tilde{y}_i, x_i) = p(y_i|\tilde{y}_i, x_i)$ represents the algorithm for label cleaning [12].

Advantage: It helps in the formation of a robust model as the base learner is used to pull the attributes from the dataset for noise detection resulting in an improvement in the overall performance of the classifier [12].

Disadvantage: Cleaning the dataset can prove to be an arduous task utilizing more time and resources and hence is not an efficient scalable method [13].

2.3 Importance Reweighting

It is based on models that learn the model weights and the noise distribution parameters simultaneously while assuming prior information regarding the label noise [14]. In order to detect the label noise, it learns the model weights and noise distribution simultaneously [14]. The empirical risk $\hat{R}_{l,D}(f)$ is shown in Eq. (6).

$$\hat{R}_{l,D}(f) = \frac{1}{N}\sum_{i=1}^{N}\beta(x_i, y_i)l(f_\theta(x_i), (\tilde{y}_i)), \tag{6}$$

where $\beta(x_i, y_i)$ determines the instance dependant weight [14].

Advantage: It results in the estimation of noise rates which further help in building more accurate model [14].

Disadvantage: It works well in the existence of both clean and noisy data, but real-world datasets always don't have clean data. Also, the necessity to fit the model two times results in more time consumption [4].

2.4 Label noise robust model

It is based on methods which are inherently more robust to label noise [15]. The first method examines different ML architectures and models robustness to label noise [3]. Different ML architecture are unequally sensitive to label noise [3]. The second model focusses on the robustness of the model depending on the type of loss functions, i.e. the robustness of label noise in the context of empirical risk minimization [3]. Mean absolute error loss, cross entropy with abstention, correcting standing loss functions using label using loss noise statistics are some of the key aspects of this approach [3].

Previous studies show that if the equation mentioned in Eq. (7) is satisfied by the loss function, then it is robust to uniform noise.

$$\sum_{k} l(f_\theta(x), k) = C, \forall x \in X, \tag{7}$$

where C is a constant value.

Advantage: It proves to be useful when no portion of the data is clean and all the samples are given similar weightage [15].

Disadvantage: It has a negative impact on noise as sometimes it is unable to differentiate between clean and noisy data [4].

3 Methods

In this section, there is a detailed description of the classification methods used in the experiment including the cost function, it's theoretical foundations and the optimization methods. Additionally, this section describes the methods used for the formulation of transition matrix and it's theoretical foundations accompanied by it's optimization algorithms.

3.1 Exploratory Data Analysis

A thorough understanding of the datasets is utmost necessary before using it to train any machine learning model. Thus, all the three datasets were analysed in order to find the distribution of features, labels and other statistical implications. The datasets were reshaped in the following order: size, channel, height and width.

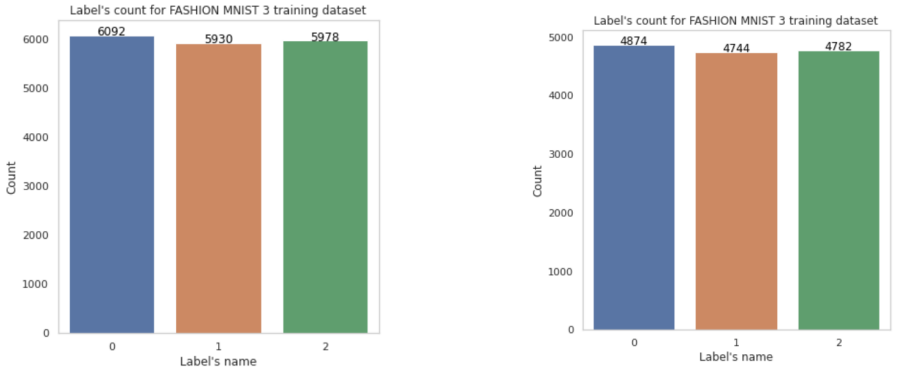
Table 1 lists the summary of the datasets.

After exploring both the test and training data of FashionMINIST0.3 and FashionMINIST0.6, it was observed that the label count in the training data was not distributed evenly whereas the label count for the test data was. On the other hand, CIFAR had even label distribution across the training and

Table 1: Exploratory data analysis of FashionMINIST0.3, FashionMINIST0.6 and CIFAR

Attributes	**FashionMINIST0.3**	**FashionMINIST0.6**	**CIFAR**
Number of labels	3	3	3
Number of images (training data)	14400	14400	12000
Number of images (validation data)	3600	3600	3000
Number of images (test data)	3000	3000	3000
Size of each image	28 X 28	28 X 28	32 X 32
Dimension of training dataset	(14400, 1, 28, 28)	(14400, 1, 28, 28)	(12000, 3, 32, 32)
Dimension of validation dataset	(3600, 1, 28, 28)	(3600, 1, 28, 28)	(3000, 3, 32, 32)
Dimension of test dataset	(3000, 1, 28, 28)	(3000, 1, 28, 28)	(3000, 3, 32, 32)
Dimension of labels (train data)	(14400,)	(14400,)	(12000,)
Dimension of labels (validation data)	(3600,)	(3600,)	(3000,)
Dimension of labels (test data)	(3000,)	(3000,)	(3000,)
Minimum pixel value	0	0	0
Maximum pixel value	255	255	255

test data. Training data was further split such that 80% of it was randomly sampled as training data and the remaining 20% as validation data. Fig. 1(a) shows the distribution of lables in the training dataset before it was split into training and validation dataset and Fig. 1(b) shows the result of stratified data split.



(a) Before splitting into training and validation data (b) After stratified split into training and validation data

Figure 1: Label distribution of FashionMINIST0.3 training dataset

3.2 Pre-processing

The pre-processing of the datasets was done to evaluate its impact while executing the experiments. The details has been documented in section 4. In order to perform pre-processing of the datasets, two scaling techniques, standardization and normalization, were used. Since the CIFAR dataset was having 3 channels, conversion to grayscale was also tried to reduce the complexity of the dataset while training it. However, it was not used later on during the experiments since Reset model was chosen for it and that's more optimal for 3 channels.

3.2.1 Standardization

This method standardizes [16] the image pixels for the given dataset for mean value 0 and standard deviation 1. The mean and standard deviation value obtained from training data were stored to standardize validation/test dataset. Fig. 2 shows 10 sample images from FashionMINIST0.6 dataset after standardization.

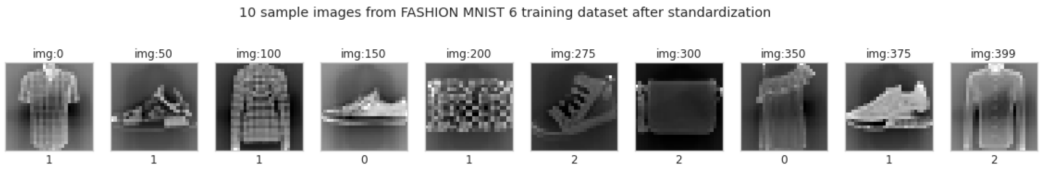


Figure 2: 10 sample images from FashionMINIST0.6 dataset after standardization

3.2.2 Normalization

This method [17] was applied to scale the pixels of the image to have value between 0 to 1. This technique helps in reducing the occurrence of gradient explosion during the training of a deep network. Fig. 3 shows 10 sample images from CIFAR dataset after normalization.

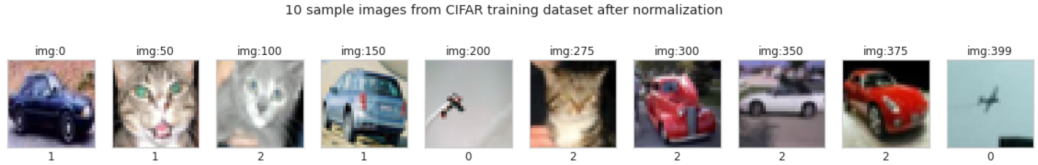


Figure 3: 10 sample images from CIFAR dataset after normalization

3.3 Cross-entropy

As it was a multi class classification problem, Cross entropy loss with Softmax was used as the last layer in the deep models to conduct the experiments. Details of the steps involved in this process are given in section 4.

The formula of Softmax activation function [18] is as given below in Eq. (8):

$$\sigma(\overrightarrow{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \tag{8}$$

where:
σ = Softmax activation function,
$\vec{z}$ = input matrix,
K = number of classes and
j = classes in the dataset.

The formula of cross-entropy loss function [19] is as given below in Eq. (9):

$$H(p) = -\sum_{x} p(x) \log p(x), \tag{9}$$

where x is the random variable and p(x) is the probability of x.

In general cross-entropy alone is not robust enough to deal with label noise [20]. As the datasets used in the experiments contained noisy labels, other methods like Forward Learning and Early Learning Regularization were used to deal with it.

3.4 Forward Learning

Forward learning plays an important role in finding the clean class posterior, assuming the transition matrix is given [21]. A relationship can be derived between the clean class posterior and noisy class posterior using forward learning using the Eq. (10). This implies that the transition matrix T when multiplied by the clean class posterior $P(Y = 1|X)$ gives a noisy class posterior $P(\tilde{Y} = 1|X)$.

$$[P(\tilde{Y} = 1|X), ..., P(\tilde{Y} = C|X)]^T = T[P(Y = 1|X), ..., P(Y = C|X)]^T. \tag{10}$$

In the traditional deep learning model, with Softmax as the activation function and cross-entropy as the loss function, when the input data had noisy labels, the output after Softmax was the approximation of noisy class posterior.

$$P(\tilde{Y}|X) = g(X) = Tq(X). \tag{11}$$

On modifying the traditional deep learning model, the transition matrix was added after the Softmax layer and it was used to calculate loss along with the cross-entropy loss function. The final output would still be approximate of noisy class posterior. This implied that the output delivered by Softmax layer when multiplied by transition matrix was a close approximation of the clean class posterior because the Eq. (11) holds true. The predictions delivered by Softmax are reliable predictions, hence this method is called forward learning. Noise transition matrix is a significant component of this method and it is discussed in detail in section 3.4.2.

3.4.1 Transfer Learning

Let us assume that X is an observation (feature), Y is the clean but unobservable label, and $\tilde{Y}$ is the observed noisy label. Suppose we have a clean training sample S such that S= (X_1, Y_1) ,..., (X_n, Y_n). When it gets contaminated by noise, it results in noisy sample $\bar{S}$ [22]:

$$\bar{S} = (x_i, \bar{y_i})_{i=1}^n. \tag{12}$$

Our objective is to build two robust classifiers from the noisy training sample that have the capability to render clean labels for the test data. A transition matrix plays a significant role in these methods by linking the revised probabilities of the class for clean and noisy data [22]. These methods leverage the noisy class posterior probabilities in the noise transition matrix to build an accurate classifier. For multi-label classification problems, the transition matrix can be estimated by leveraging the anchor points [21]. A transition matrix $T(x)$ is defined as a function of matrix values which is used in a process to render label noise [21].

The transition matrix T can be defined as [21]:

$$T = \begin{bmatrix} P(\tilde{Y} = 1|Y = 1) & P(\tilde{Y} = 1|Y = 2) & .... & P(\tilde{Y} = 1|Y = C) \\ P(\tilde{Y} = 2|Y = 1) & P(\tilde{Y} = 2|Y = 2) & .... & (\tilde{Y} = 2|Y = C) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ P(\tilde{Y} = C|Y = 1) & P(\tilde{Y} = C|Y = 2) & .... & (\tilde{Y} = C|Y = C) \end{bmatrix}, \tag{13}$$

where $P(\tilde{Y} = 1|X)$ is the noisy class posterior and C is the class label.

3.4.2 Estimating the transition matrix using the anchor points

Anchor points are the maximum values (probabilities) defined in the clean data set [21]. An instance x in the dataset, can be defined as an anchor point for class i if the instance has a probability equal to 1 in the i^{th} class [21] as shown in Eq. (14). This anchor point (x) will contain the most discriminative information about the i^{th} class [21].

$$P(Y = i|X = x) = 1. \tag{14}$$

This further helps in constructing the transition matrix T by estimating the revised probabilities for anchor points as shown in Eq. (15) [21]:

$$\begin{bmatrix} P(\tilde{Y}=1|X) \\ . \\ . \\ . \\ P(\tilde{Y}=C|X) \end{bmatrix} = \begin{bmatrix} P(\tilde{Y}=1|Y=1) & ... & P(\tilde{Y}=1|Y=C) \\ . & & \\ . & & \\ . & & \\ P(\tilde{Y}=C|Y=1) & ... & P(\tilde{Y}=C|Y=C) \end{bmatrix} \begin{bmatrix} P(Y=1|X) \\ . \\ . \\ . \\ P(Y=C|X) \end{bmatrix}. \quad (15)$$

This equation can be obtained by the using the law of total probability [22]. The noisy class posterior can be rewritten as the product of transition matrix and clean class posterior [22]. This transition matrix can be estimated by exploiting the anchor point [22].

$$[P(Y=1|X=x^1)=1, P(Y=2|X=x^1)=0, ..., P(Y=C|X=x^1)=0]^T. \quad (16)$$

Assuming we have x^1 as the anchor point from the first class, it can be observed that the anchor point is equal to the first column of the transition matrix [22]. This can be generalised that if the anchor points for the i^{th} class is given, then the i^{th} column of the transition matrix can be learnt. The equation is given as follows:

$$\begin{bmatrix} P(\tilde{Y}=1|X=x^i) \\ . \\ . \\ . \\ P(\tilde{Y}=C|X=x^i) \end{bmatrix}, \quad (17)$$

where x^i is the anchor point of the i^{th} class.

3.4.3 Estimation of anchor points

In situations where anchor point is provided, the transition matrix can be calculated easily. In case the anchor points are not already given, they need to be calculated. However, the anchor points are unknown in real world datasets and are needed to be identified either using a theoretical approach or a practical hands-on approach [23]. They are based on the assumptions that the matrix is symmetric, and the flip rates of instances are upper bounded i.e. the flip rates are so small that and are always non-negative [23]. We can find anchor point for the i^{th} class by finding with the highest noisy class posterior [23]. The intent is to treat noisy class posteriors with high value as anchor points for the i^{th} class [23]. This approach involves some approximation error, because we are estimating anchors points from noisy data but it also helps in building a classifier robust to noisy labels [23].

The transition matrix of FashionMINIST0.3 and FashionMINIST0.6 was already provided but the transition matrix had to be estimated for the CIFAR dataset. It was calculated using anchor points method. The details are described in section 4.

3.5 Early Learning Regularization

This was the second approach used in the experiments which uses deep learning models to perform classifications in noisy labelled datasets. Previous studies showed that during the initial training stages, when supplied with data with noisy labels, the model tends to first fit the clean training data before memorizing the data with noisy labels [26]. This phenomenon occurs because the gradients associated with the clean labels govern the fundaments of the training process resulting in faster advancement towards the true optimal values [26]. Soon enough, the gradients associated with noisy data starts taking over the dynamics of the training process and by that time the model has already learnt to fit the noisy labels [26]. This eventually leads to miss-classifications during the test phase and weak performance of the classifier [26].

In order to learn clean labels and refrain from memorization, it is important to assure there is significant contribution of gradients of data with clean labels [26]. Apart from that, the effect the noisy labels must be balanced [26].

Using this approach, the advancement of the early learning phase is tracked through regularization in order to classify noisy labels [26]. This is different from the regular approaches because this

technique does not involve altering the labels, instead the regularization term focusses on rectifying the gradient of the loss function [26]. This method takes advantage of it's two supporting factors: One of them is to exploit the semi-supervised learning methods to find target probabilities for each training set (as shown in Eq. (18)) [26].

The other factor is to construct a regularization term which drives the classifier heading in the direction of the target probabilities [26].

$$L_{ELR}(\Theta) = L_{CE}(\Theta) + \frac{\lambda}{n}\sum_{i=1}^{n}\log(1- < p^{[i]}, t^{[i]} >), \tag{18}$$

where:
θ represents the parameters of the deep learning model,
p denotes the Softmax activation function,
t denotes the target vector,
L_{ELR} is the early learning regularization loss function, and
L_{CE} is the cross entropy loss function [26].

$$t^{[i]}(k) = \beta t^{[i]}(k-1) + (1-\beta)p^{[i]}(k), \tag{19}$$

where:
$t^{[i]}(k)$ is the target vector,
$p^{[i]}(k)$ is the output delivered by the model and
β lies between 0 to 1 and is the running momentum [26].

Advantage: This generates good performance and high accuracy without the need of selecting the sample data [26]. This approach can be used for classification as well as for deep learning models.

Disadvantage: The target vector required in order to implement Early Learning Regularization scales in complexity with the increase in the size of the dataset. When compared to DivideMix, the empirical performance did not prove to be good enough [30].

4 Experiments

Three datasets namely FashionMINIST0.3, FashionMINIST0.6 and CIFAR were given as part of this assignment. The datasets have been explained in detail in section 3.1.
Top-1 accuracy and loss were used as the evaluation metrics to analyse the performance of the experiments. Each experiment was run 10 times and the average top-1 accuracy, average loss and the standard deviation of the same was calculated.
In each run, the data was split into training and val data using random sampling of 80% and 20% respectively. The training data of FashionMINIST0.3 and FashionMINIST0.6 was not completely balanced for the 3 target labels. So in order to maintain the same ratio of data in each random sampling of the training and validation data split, stratified sampling technique was used.

Since the labels of the training datasets were noisy, the following steps were used to analyse and address the issue.

- A multi-class classifier was designed using a neural network having Softmax and cross entropy loss.
- Hyper parameter tuning of the classifier was done to fit the training data well. Many parameters like different network architecture (MLP, ResNet, LeNet) with dropout, batch normalization, changing different batch size, using various pre-processing techniques (normalization, standardization), trying various learning rates, different optimizer techniques (SGD, Adam, Nesterov) and their parameter's combination and scheduler were tried and then the final setup which worked best for each type of dataset was used to continue with further experiments. various numbers of epochs were also used to understand what should be the optimal number for it. Refer to section 8 in the code which was used to conduct these hyper-parameters tuning.

- Transition matrix was estimated based on the trained model.
- The Forward Learning method with transition matrix was used on the chosen model to understand its impact on label noise.
- Another technique namely Early Learning Regularization was also tried as the second method to analyse its performance on datasets having label noise.

4.1 Experiments on FashionMINIST0.3 and FashionMINIST0.6

FashionMINIST0.3 and FashionMINIST0.6 had gray-scale images. All the models like MLP, modified ResNet (to work with a single channel of gray-scale image), and LeNet had almost similar performance on these two datasets. LeNet was chosen to perform further experiments since it is geared towards gray-scale images. Table 2 and Table 3 captures the summary of the experiment done on FashionMINIST0.3 and FashionMINIST0.6 respectively. It is clear that the accuracy of Forward Learning and Early Learning Regularization was very close but there was a good difference in the loss for both the methods in both datasets.

Fixed Parameters
Batch Size - 64
Pre-processing technique - Normalization
Model - LeNet with dropout and batch normalization
Repeat - 10 times
Epochs - 50

Table 2: Experiment summary for FashionMINIST0.3 dataset

FashionMINIST0.3 dataset						
Method	**Avg Train Top-1 Acc (max)**	**Avg Train Loss (min)**	**Avg Val Top-1 Acc (max)**	**Avg Val Loss (min)**	**Test Top-1 Acc**	**Test Loss**
Cross Entropy (CE)	67.81	0.66	69.04	**0.64**	**98.63**	0.40
Forward Learning (FWD)	69.13	0.62	**69.44**	2.75	98.5	**0.04**
Early Learning Regularization (ELR)	**69.14**	**-21.06**	69.21	1.00	97.86	0.28

Table 3: Experiment summary for FashionMINIST0.6 dataset

FashionMINIST0.6 dataset						
Method	**Avg Train Top-1 Acc (max)**	**Avg Train Loss (min)**	**Avg Val Top-1 Acc (max)**	**Avg Val Loss (min)**	**Test Top-1 Acc**	**Test Loss**
Cross Entropy (CE)	36.65	1.10	38.32	**1.09**	82.83	0.99
Forward Learning (FWD)	**38.79**	1.09	**38.70**	6.60	**94.79**	**0.20**
Early Learning Regularization (ELR)	38.74	**-16.14**	38.24	1.40	87.66	0.73

Fig. 4 and Fig. 5 displays the average and standard deviation for Top-1 accuracy and loss captured for Cross Entropy(CE), Forward Learning(FWD) and Early Learning Regularization(ELR) on FashionMINIST0.3 and FashionMINIST0.6 dataset respectively.

In Fig. 4, it is clear that the accuracy of Forward Learning and Early Learning Regularization was better than only Cross Entropy. The loss of training data using Early Learning Regularization was very low whereas for validation data, it was higher than the other methods. The standard deviation for accuracy was high for Cross Entropy validation data but otherwise stable for other methods. The standard deviation for loss was higher for Early Learning Regularization training data followed by validation data. It was also fairly stable for other methods.

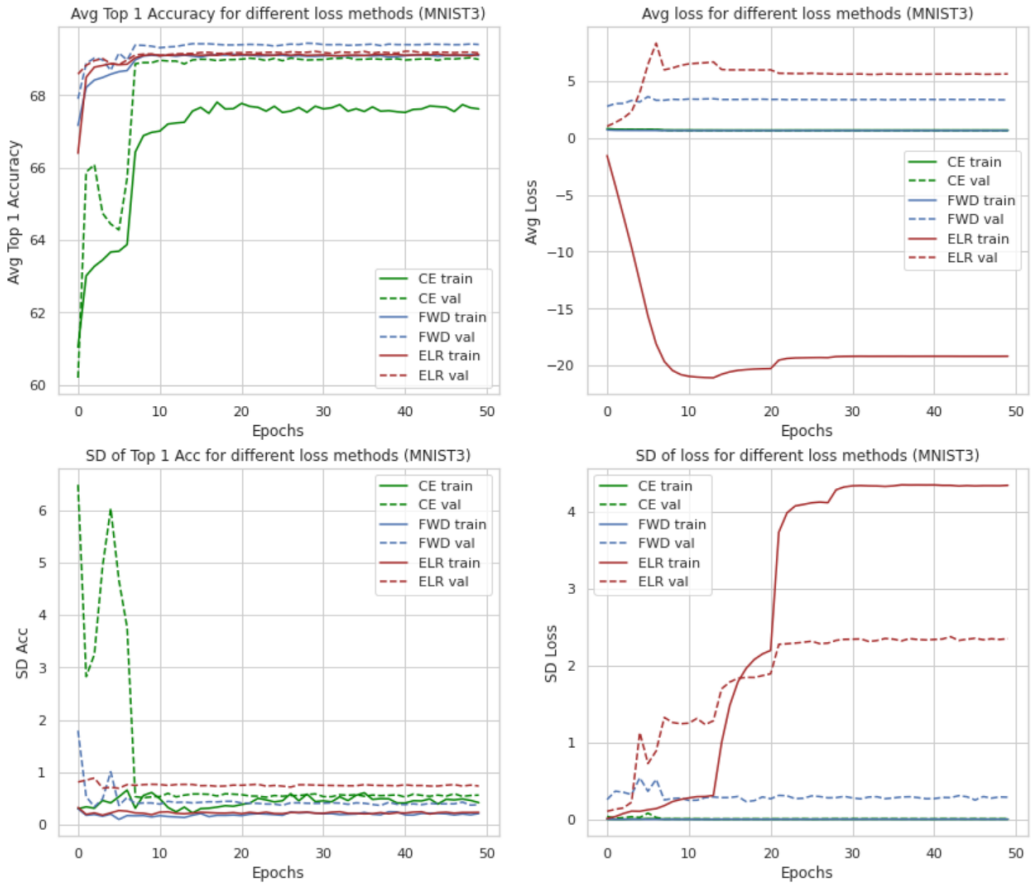


Figure 4: Average and Standard deviation for Top-1 Accuracy and Loss captured for Cross Entropy(CE), Forward Learning(FWD) and Early Learning Regularization(ELR) on FashionMINIST0.3 dataset

In Fig. 5, it is clear that the accuracy of Forward Learning and Early Learning Regularization was better than only Cross Entropy. The loss of training data using Early Learning Regularization was very low whereas for validation data, it was higher than the other methods but for validation of Forward Learning method which also had comparatively high validation loss. The standard deviation for accuracy for validation dataset for all the methods was having a slight variation but was almost stable for the training datasets. The standard deviation for loss was higher for Early Learning Regularization training data followed by validation data of its own and Forward Learning.

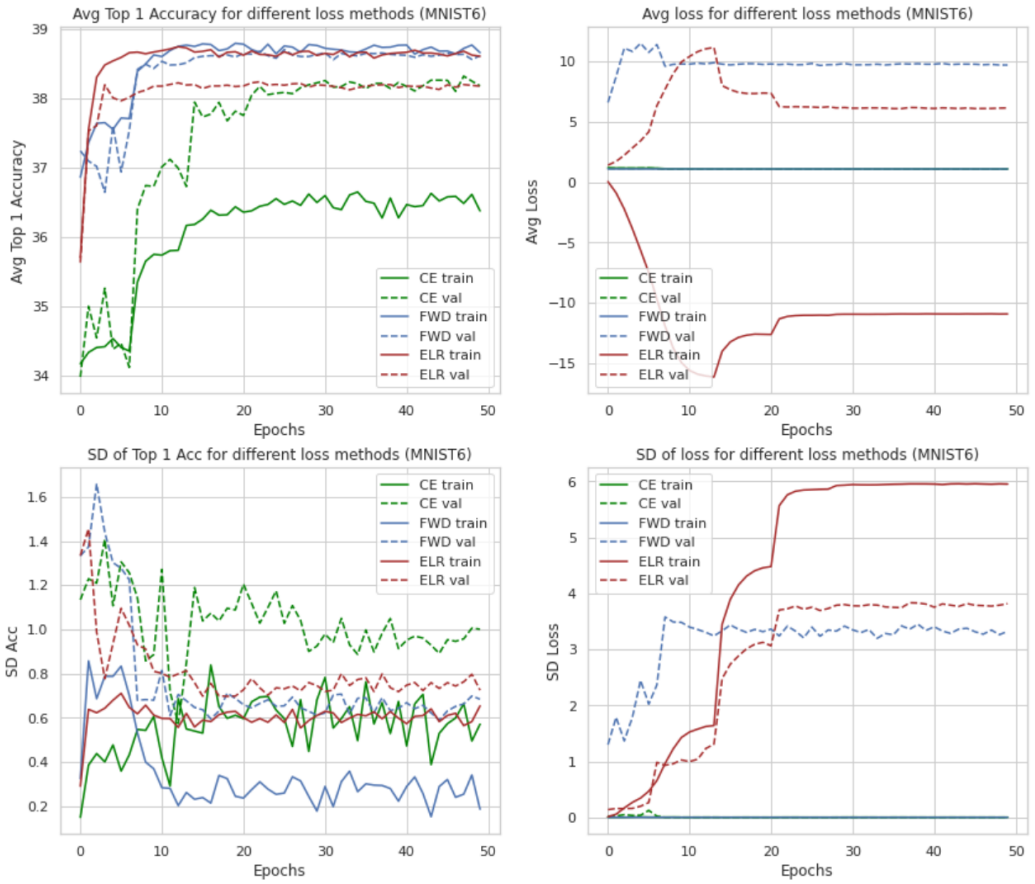


Figure 5: Average and Standard deviation for Top-1 Accuracy and Loss captured for Cross Entropy(CE), Forward Learning(FWD) and Early Learning Regularization(ELR) on FashionMINIST0.6 dataset

The transition matrix for FashionMINIST0.3 and FashionMINIST0.6 dataset was given as part of the assignment. The transition matrix for both datasets were estimated using the transition matrix estimator written for the CIFAR dataset. The estimated transition matrix for FashionMINIST0.3 and FashionMINIST0.6 were compared with the original one and the error was calculated by taking the sum of the absolute difference of the values of the two matrices and dividing it by the sum of the values of the original transition matrix [29].
Table 4 shows the original and estimated transition matrix and the error between the two. It is clear that the estimated transition matrix was very close to the original ones. The estimated one for FashionMINIST0.6 was better estimated than that of FashionMINIST0.3.

Table 4: Transition Matrix comparison for FashionMINIST0.3 and FashionMINIST0.6 datasets

Dataset	**Original Matrix**	**Estimated Matrix**	**Error**
MINIST 3	[[0.7000, 0.3000, 0.0000], [0.0000, 0.7000, 0.3000], [0.3000, 0.0000, 0.7000]]	[[8.3091e-01, 1.6873e-01, 3.6007e-04], [2.0871e-03, 8.0663e-01, 1.9128e-01], [1.4923e-01, 1.3354e-03, 8.4943e-01]]	0.2605
MINIST 6	[[0.4000, 0.3000, 0.3000], [0.3000, 0.4000, 0.3000], [0.3000, 0.3000, 0.4000]]	[[0.4191, 0.2824, 0.2985], [0.3088, 0.4095, 0.2818], [0.2690, 0.2911, 0.4399]]	0.0515

4.2 Experiments on CIFAR

CIFAR dataset had coloured images. ResNet18 was chosen to perform further experiments since it is geared towards coloured images. ResNet50 was also tried with dropout and batch normalization but it resulted in intermittent gradient explosion. Fine tuning of various parameters did help in reducing the occurrence of the issue, but it was resolved with using ResNet18 which involves less computation than ResNet50.
Table 5 captures the summary of the experiment done on CIFAR. It is clear that the on training data, the accuracy of Forward Learning was better than the other methods. The loss on training data for Early Learning Regularization was very low. The validation data's accuracy and loss were very close for all the methods. The test data's accuracy was highest for Early Learning Regularization method but test loss was also highest in this case whereas it was comparable for Forward Learning and Cross Entropy alone.

Fixed Parameters
Batch Size - 64
Pre-processing technique - Normalization
Model - ResNet18
Repeat - 10 times
Epochs - 30

Table 5: Experiment summary for CIFAR dataset

CIFAR dataset						
Method	**Avg Train Top-1 Acc (max)**	**Avg Train Loss (min)**	**Avg Val Top-1 Acc (max)**	**Avg Val Loss (min)**	**Test Top-1 Acc**	**Test Loss**
Cross Entropy (CE)	55.85	0.91	35.33	**1.15**	48.69	**1.01**
Forward Learning (FWD)	**58.83**	0.87	35.63	1.17	37.03	1.11
Early Learning Regularization (ELR)	42.27	**-13.16**	**37.28**	1.36	**55.36**	6.12

Fig. 6 displays the average and standard deviation for Top-1 accuracy and loss captured for Cross Entropy(CE), Forward Learning(FWD) and Early Learning Regularization(ELR) methods on CIFAR dataset.
It is evident that the accuracy of training data was better than that of validation data for all the methods.The accuracy of Forward Learning was best on the training data. Early Learning Regularization plateaued very fast for the training data's accuracy. The loss of training data using Early Learning Regularization was very low whereas for validation data, it was higher than the other methods. The standard deviation for accuracy of training dataset for Forward Learning and Cross Entropy was having more variation than that of Early Learning Regularization and validation datasets in general. The standard deviation for loss was having some variation for Early Learning Regularization training data followed by validation data. It was fairly stable for the other datasets and methods.

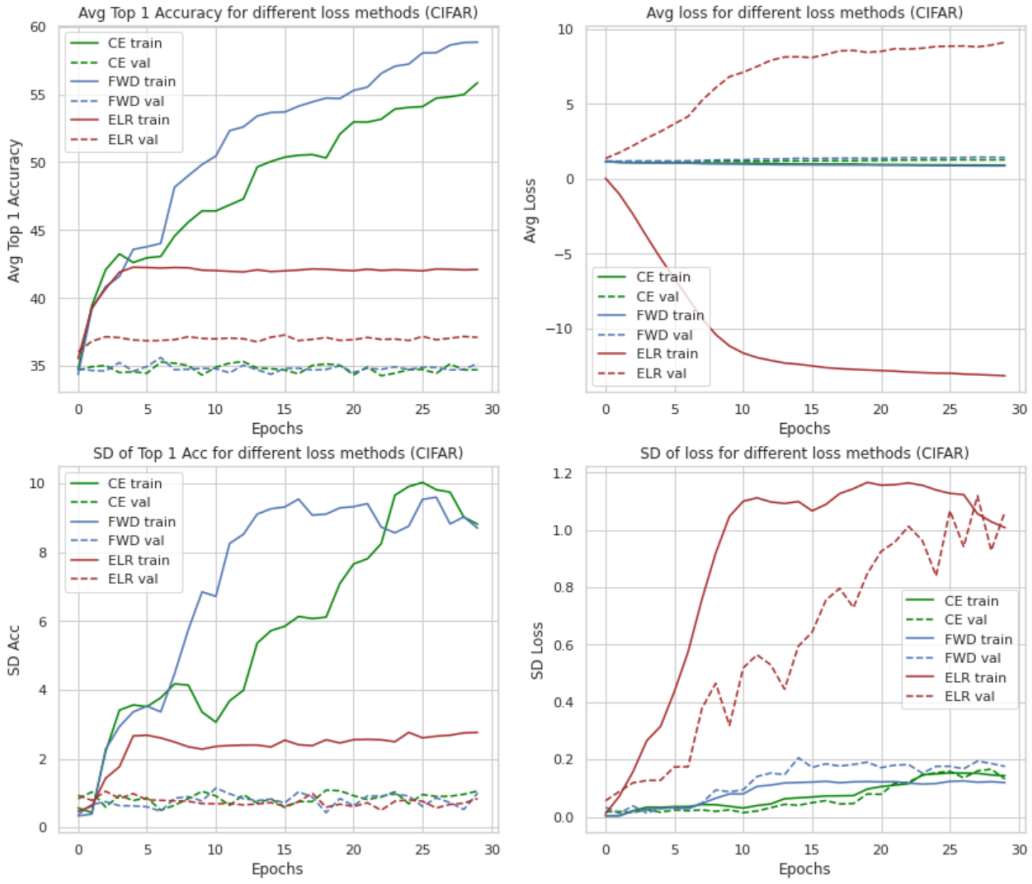


Figure 6: Average and Standard deviation for Top-1 Accuracy and Loss captured for Cross Entropy(CE), Forward Learning(FWD) and Early Learning Regularization(ELR) on CIFAR dataset

The transition matrix for CIFAR dataset was not given. The estimated transition matrix is shown in Fig. 7.

```
Estimated transition matrix
:  tensor([[0.8035, 0.1199, 0.0767],
        [0.0101, 0.9411, 0.0489],
        [0.0255, 0.0340, 0.9405]])
```

Figure 7: Estimated Transition matrix for CIFAR dataset

5 Conclusion

5.1 Conclusion

- There was overall improvement in classification result after adding Forward Learning and Early Learning Regularization.
- The loss was very low and negative number for training data in case of Early Learning Regularization.
- ResNet50 with dropout and batch normalization resulted in intermittent gradient explosion with a lot of setting's configurations. Default ResNet18 had similar performance but due to being simple than ResNet50, it was more stable.
- Normalized images had better performance than standardized images.
- Increasing the batch size did help to improve the performance but after it was increased to more than 64, it didn't improve any further.
- Fine tuning learning rate and optimizer did help in avoiding gradient explosion problem.

5.2 Future Work

FashionMINIST0.3 had better performance than the other two datasets. More fine tuning of hyper-parameters could have resulted in better performance. But this is a very time intensive process and due to the limit of time and scope of this project, this could be taken as a future work.

Apart from this, efficient methods for dealing with label noise are very important as sensitive and large datasets have labels that are unique and have greater weight than multiple features with varied importance in a particular dataset. Given the impact that learning with label noise has on the datasets, there is a huge potential of research in multiple areas:

- Besides focusing on various categories of label noise, future research topics should develop a comprehensive apprehension of the impact of label noise on deep neural networks.
- Studies should focus on detecting which part of convolutional layers (initial or last) are affected by label noise. Additionally, learning with label noise can be used to detect the section of the network that is affected through transfer learning.
- Researchers can also focus on finding methods to train data in the presence of both feature and label noise. Moreover, some work can be done to understand the layout of noise for distributing it evenly across the attributes of the dataset.

6 Appendix

6.1 System Configuration

- Hardware requirements- The code was run on Google Colab on GPU runtime.
- Software requirements- Windows 10, Python 3

6.2 Instructions to execute the code

The code was developed and tested on google Colab on GPU runtime. The name of the file is 500657898_510282930.ipynb. Follow the steps mentioned below to execute the code:

- Execute all the cells in section 1.1 Import namespaces.
- Update the dataset file paths in the 3rd cell of section 1.2 Define File Paths.
- Execute all the cells in section 1.2 Define File Paths.
- Google authentication is needed to connect to google drive to access the datasets if they are kept there.
- Execute section 1.3 and 1.4 to load the datasets.

- Execute all cells in section 2 for Initial data analysis.
- Execute all cells in section 3 for splitting training and validation data.
- Execute all cells in section 4 for Pre-Processing techniques.
- Execute all cells in section 5 for defining and estimating transition matrix.
- Execute all cells in section 6 for model, classification algorithm setup.
- Execute all cells in section 7 for result plotting code.
- Section 8 can be skipped. This section has code to do fine tuning of the model on each dataset.
- Execute all cells in section 9 for Experiment setup.
- Execute all cells in section 10 for Experiment setup execution.

6.3 Contribution of the team members

Both the team members contributed equally to the assignment.

7 References

[1]H. Song, M. Kim, D. Park and J. Lee, "Learning from noisy labels with deep neural networks: A survey.", CoRR, vol. 200708199, 2020. Available: https://arxiv.org/abs/2007.08199. [Accessed 5 November 2021].

[2]B. Frenay and M. Verleysen, "Classification in the Presence of Label Noise: A Survey", IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 5, pp. 845-869, 2014. Available: 10.1109/tnnls.2013.2292894 [Accessed 6 November 2021].

[3]SimilarWeb, PyData Tel Aviv Meetup: Shaky Ground (truth): Learning with Label Noise. 2019.

[4]G. Algan and I. Ulusoy, "Image classification with deep learning in the presence of noisy labels: A survey", Knowledge-Based Systems, vol. 215, p. 106771, 2021. Available: 10.1016/j.knosys.2021.106771 [Accessed 10 November 2021].

[5]N. Natarajan, I. Dhillon, P. Ravikumar and A. Tewari, "Learning with noisy labels", Advances in neural information processing systems, vol. 26, pp. 1196–1204, 2013. Available: https://proceedings.neurips.cc/paper/2013/file/3871bd64012152bfb53fdf04b401193f-Paper.pdf. [Accessed 7 November 2021].

[6]G. Patrini, A. Rozza, A. Menon, R. Nock and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach", Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1944–1952, 2017. Available: https://arxiv.org/pdf/1609.03683.pdf. [Accessed 3 November 2021].

[7]A. Berthon, B. Han, G. Niu, T. Liu and M. Sugiyama, "Confidence scores make instance-dependent label-noise learning possible", International Conference on Machine Learning, pp. 825–836, 2021. Available: https://arxiv.org/pdf/2001.03772.pdf. [Accessed 6 November 2021].

[8]D. Hendrycks, K. Lee and M. Mazeika, "Using Pre-Training Can Improve Model Robustness and Uncertainty", CoRR, pp. 2712-2721, 2019. Available: http://arxiv.org/abs/1901.09960. [Accessed 4 November 2021].

[9]J. Han, P. Luo and X. Wang, "Deep Self-Learning From Noisy Labels", Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 5138-5147, 2019. Available: https://openaccess.thecvf.com/content_ICCV_2019/html/Han_Deep_Self-Learning_From_Noisy_Labels_ICCV_2019_paper.html. [Accessed 5 November 2021].

[10]G. Algan and I. Ulusoy, "Image classification with deep learning in the presence of noisy labels: A survey", Knowledge-Based Systems, vol. 215, p. 106771, 2021. Available: https://arxiv.org/pdf/1912.05170.pdf.

[11]"Robust Loss Functions under Label Noise for Deep Neural Networks", Thirty-First AAAI Conference on Artificial Intelligence, vol. 31, no. 1, 2017. Available: https://ojs.aaai.org/index.php/AAAI/article/view/10894. [Accessed 8 November 2021].

[12]Y. Zhang, W. Deng, M. Wang, J. Hu, D. Zhao and D. Wen, "Global-Local GCN: Large-Scale Label Noise Cleansing for Face Recognition", Proceedings of the IEEE/CVF

Conference on Computer Vision and Pattern Recognition, pp. 7731–7740, 2020. Available: https://openaccess.thecvf.com/content_CVPR_2020/papers/Zhang_Global-Local_GCN_Large-Scale_Label_Noise_Cleansing_for_Face_Recognition_CVPR_2020_paper.pdf. [Accessed 7 November 2021].

[13]J. Krause et al., "The unreasonable effectiveness of noisy data for fine-grained recognition", CoRR, 2015. Available: https://arxiv.org/abs/1511.06789. [Accessed 5 November 2021]

[14]T. Liu and D. Tao, "Classification with Noisy Labels by Importance Reweighting", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 3, pp. 447-461, 2016. Available: 10.1109/tpami.2015.2456899 [Accessed 6 November 2021].

[15]X. Yu, T. Liu, M. Gong, K. Zhang, K. Batmanghelich and D. Tao, "Label-Noise Robust Domain Adaptation", Proceedings of the 37th International Conference on Machine Learning, pp. 10913–10924, 2020. Available: http://proceedings.mlr.press/v119/yu20c.html. [Accessed 4 November 2021].

[16]L. Buitinck et al., "API design for machine learning software: experiences from the scikit-learn project", ECML PKDD Workshop: Languages for Data Mining and Machine Learning, pp. 108–122, 2013. Available: https://scikit-learn.org/stable/modules/preprocessing.html. [Accessed 7 November 2021].

[17]F. Pedregosa et al., "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research, vol. 12, no. 85, pp. 2825-2830, 2011. Available: http://jmlr.org/papers/v12/pedregosa11a.html. [Accessed 6 November 2021].

[18]H. Mahmood, "Softmax Function, Simplified", Medium, 2018. [Online]. Available: https://towardsdatascience.com/softmax-function-simplified-714068bf8156. [Accessed: 05- Nov- 2021].

[19]J. Brownlee, "A Gentle Introduction to Cross-Entropy for Machine Learning", Machine Learning Mastery, 2020. [Online]. Available: https://machinelearningmastery.com/cross-entropy-for-machine-learning/. [Accessed: 04- Nov- 2021].

[20]L. Feng, S. Shu2, Z. Lin1, F. Lv, L. Li and B. An, "Can cross entropy loss be robust to label noise?", IJCAI, pp. 2206–2212, 2020. Available: https://www.ijcai.org/proceedings/2020/0305.pdf. [Accessed 5 November 2021].

[21]X. Xia et al., "Are anchor points really indispensable in label-noise learning?", Advances in Neural Information Processing Systems, vol. 32, pp. 6838–6849, 2019. Available: https://arxiv.org/pdf/1906.00189.pdf. [Accessed 6 November 2021].

[22]X. Xia et al., "Part-dependent Label Noise: Towards Instance-dependent Label Noise", Advances in Neural Information Processing Systems, vol. 33, 2020. Available: https://proceedings.neurips.cc/paper/2020/file/5607fe8879e4fd269e88387e8cb30b7e-Paper.pdf. [Accessed 6 November 2021].

[23]B. Han et al., "A survey of label-noise representation learning: Past, present and future", arXiv preprint arXiv:2011.04406, 2020. Available: https://arxiv.org/pdf/2011.04406.pdf. [Accessed 6 November 2021].

[24]M. Ciortan, R. Dupuis and T. Peel, "A Framework Using Contrastive Learning for Classification with Noisy Labels", Data, vol. 6, no. 6, p. 61, 2021. Available: 10.3390/data6060061 [Accessed 3 November 2021].

[25]K. McGuinness, "Deep Learning with Label Noise", Youtube.com, 2019. [Online]. Available: https://www.youtube.com/watch?v=8mpBHbjG4E4. [Accessed: 06- Nov- 2021].

[26]S. Liu, J. Niles-Weed, N. Razavian and C. Fernandez-Granda, "Early-Learning Regularization Prevents Memorization of Noisy Labels", vol. 200700151, 2020. Available: https://arxiv.org/pdf/2007.00151.pdf. [Accessed 7 November 2021].

[27]X. Xia et al., "Are anchor points really indispensable in label-noise learning?", Advances in Neural Information Processing Systems, vol. 32, pp. 6838–6849, 2019. Available: https://arxiv.org/pdf/1906.00189.pdf. [Accessed 6 November 2021].

[28]X. Li, T. Liu, B. Han, G. Niu and M. Sugiyama, "Provably End-to-end Label-noise Learning without Anchor Points", arXiv preprint arXiv:2102.02400, 2021. Available: https://arxiv.org/pdf/2102.02400.pdf. [Accessed 3 November 2021].

[27]X. Li, T. Liu, B. Han, G. Niu and M. Sugiyama, "Provably End-to-end Label-noise Learning without Anchor Points", arXiv preprint arXiv:2102.02400, 2021. Available: https://arxiv.org/pdf/2102.02400.pdf. [Accessed 3 November 2021].

[28]Y. Yao et al., "Dual T: Reducing Estimation Error for Transition Matrix in Label-noise Learning", arXiv preprint arXiv:2102.02400, 2021. Available:

https://proceedings.neurips.cc/paper/2020/file/512c5cad6c37edb98ae91c8a76c3a291-Paper.pdf. [Accessed 7 November 2021].

[29]X. Xia, "GitHub - xiaoboxia/T-Revision: NeurIPS'2019: Are Anchor Points Really Indispensable in Label-Noise Learning?", GitHub, 2019. [Online]. Available: https://github.com/xiaoboxia/T-Revision. [Accessed: 08- Nov- 2021].

[30]"Review for NeurIPS paper: Early-Learning Regularization Prevents Memorization of Noisy Labels", Proceedings.neurips.cc, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/ea89621bee7c88b2c5be6681c8ef4906-Review.html. [Accessed: 06- Nov- 2021].