*Computers, Materials & Continua*
DOI: 10.32604/cmc.2022.028154
***Article***

Tech Science Press

# Iterative Semi-Supervised Learning Using Softmax Probability

**Heewon Chung and Jinseok Lee***

Department of Biomedical Engineering, College of Electronics and Information, Kyung Hee University, Yongin-si, Gyeonggi-do, 17104, Korea
*Corresponding Author: Jinseok Lee. Email: gonasago@khu.ac.kr
Received: 03 February 2022; Accepted: 10 March 2022

**Abstract:** For the classification problem in practice, one of the challenging issues is to obtain enough labeled data for training. Moreover, even if such labeled data has been sufficiently accumulated, most datasets often exhibit long-tailed distribution with heavy class imbalance, which results in a biased model towards a majority class. To alleviate such class imbalance, semi-supervised learning methods using additional unlabeled data have been considered. However, as a matter of course, the accuracy is much lower than that from supervised learning. In this study, under the assumption that additional unlabeled data is available, we propose the iterative semi-supervised learning algorithms, which iteratively correct the labeling of the extra unlabeled data based on softmax probabilities. The results show that the proposed algorithms provide the accuracy as high as that from the supervised learning. To validate the proposed algorithms, we tested on the two scenarios: with the balanced unlabeled dataset and with the imbalanced unlabeled dataset. Under both scenarios, our proposed semi-supervised learning algorithms provided higher accuracy than previous state-of-the-arts. Code is available at https://github.com/HeewonChung92/iterative-semi-learning.

**Keywords:** Semi-supervised learning; class imbalance; iterative learning; unlabeled data

## 1 Introduction

Image classification is a problem to categorize images into one of the multiple classes. It has been considered one of the most important tasks since it is the basis for other computer vision tasks such as image detection, localization and segmentation [1–6]. Since AlexNet [7] was introduced, deep neural networks (DNNs) have evolved remarkably via VGG-16 [8], GoogLeNet [9], ResNet [10], Inception-V3 [11], especially to solve the image classification tasks. DNNs have been widely used for a variety of tasks and set the new state-of-the-art, sometimes even surpassing human performance on image classification tasks.

However, when dealing with the classification problem in practice, we face many practical issues, and one of the most challenging issues is acquiring enough labeled data for training. The acquisition

of the labeled data often requires a lot of time while also requiring professional and delicate works. A recent study reported that physicians spent an average of 16 minutes and 14 seconds per encounter using electronic health record (EHRs), with chart review (33%), documentation (24%), and ordering (17%) functions accounting for most of the time [12]. The manual labeling of medical images also requires intensive labor [13,14]. In addition, even if the labeled data is acquired enough, there is another challenging issue referred to as imbalanced dataset. For instance, for the classification of a specific disease data, there is much more information about the data from healthy subjects than those from patients.

To resolve these issues, semi-supervised learning methods using additional unlabeled data have been being considered a lot. Semi-supervised learning is a machine learning approach that combines a small amount of labeled data with a large amount of unlabeled data during training [15–17]. In this study, we propose a novel semi-supervised learning algorithms providing the performance at the level of supervised learning by focusing on automatically and accurately labeling additional unlabeled data. More specifically, to accurately label the unlabeled data, we use a softmax probability as a confidence index and decide whether to assign a pseudo-label to the unlabeled data. The data with labels are used continuously for training. Finally, the process is repeated until the pseudo-labels are assigned to all unlabeled data with high confidence. Our proposed approach is innovative because it effectively and accurately labels the unlabeled data using a simple mathematical function of softmax. For classification problems, softmax is essential part of a model, usually used in the last output layer. Thus, we expect to be able to effectively label the unlabeled data without additional computational complexity.

This paper is organized as follows. Section 2 lists some related works. Section 3 provides a specific motivation of dealing with unlabeled data. In Section 4, we introduce our proposed iterative semi-supervised learning using softmax probabilities. In Section 5, the performance of our algorithm is verified by comparative experiments. The conclusion and future work are described in Section 6.

## 2 Related Works

The difficulty of acquiring labeled data and the imbalanced data issue have been investigated by many research groups [18–21]. One of the popular approach to handle the imbalanced data issue is with data-level techniques including over-sampling and under-sampling [22–24]. The under-sampling is a technique to balance an imbalanced dataset by keeping all of the data in the minority group and decreasing the size of the majority group. This technique is mainly used when the amount of data belonging to minority and majority groups is large. The over-sampling is a technique to balance an imbalanced dataset by increasing the size of the minority group. This technique is mainly to duplicate minority data by randomly selecting the data from the minority group. A more advanced technique is the synthetic minority oversampling technique (SMOTE), which generates a new data point by selecting a point on a line connecting a randomly chosen minority class sample and one of its k nearest neighbors [25]. Let us denote the synthetic data point by $x_{new}$, which can be expressed as

$$x_{new} = x + \lambda \cdot (x - x_{near}) , \tag{1}$$

where $x$ is a random data belonging to a minority group, $x_{near}$ is one of the $k$ nearest neighbors of $x$. The parameter $\lambda$ is independent and identically distributed number uniformly distributed on [0,1]. This SMOTE has the advantage that of being able to increase the size of the minority group without duplicating the data. Similar to SMOTE, adaptive synthetic sampling (ADASYN) technique generates a new data point based on the k nearest neighbors [26]. It generates more data that are harder to learn compared to the data that are easier to learn by considering the data distribution. Thus, it

can adaptively shift the decision boundary to focus on the hard-to-learn data. Since the data-level techniques from over-sampling approach balance out the number of each group of data, the trained models have worked well in a variety of applications. However, such over-sampling techniques are available when the data is represented as a vector.

Another approach to handle the imbalanced data issue is with algorithmic methods. In the algorithmic approach, the learning process is adjusted in a way that emphasizes the importance of the minority group data. Most commonly, the cost or loss function is modified to weigh more towards the minority group data or to weigh less towards the majority group data [18,27,28]. Such a sample weighting in loss function is to weigh the loss computed for different samples differently based on whether they belong to the majority or the minority group. For the weight factors, inverse of number of samples or inverse of square root of the number of samples can be considered. Recently, Cui et al. [29] introduced the effective number of samples $E_{n_c}$, which can be defined as

$$E_{n_c} = \frac{1-\beta_{n_c}}{1-\beta}, \tag{2}$$

where $n_c$ is the number of samples in class $c$ and $\beta$ is a hyperparameter on [0,1]. By using the effective number of samples, the weight factor $1/E_{n_c}$ weigh the loss from the data according to the majority or the minority group. This algorithm approach also worked well in a variety of applications. Nevertheless, the imbalanced dataset issue is not completely solved. The fundamental solution is to increase the number of data with diversity by acquiring more new data.

As we mentioned above, the most challenging part of acquiring data is labeling new data. It not only takes a lot of time, but also requires professional and delicate works. Recently, Yang et al. [30] demonstrated that pseudo-label on extra unlabeled data can improve the classification performance, especially with the imbalanced dataset. The method is based on the fact that the unlabeled data is relatively easy to obtain while the labeled one is difficult to obtain. Based on the trained model with original data, extra unlabeled data was subsequently labeled. Accordingly, it was shown that the trained model with additional unlabeled data provided better performance. However, the pseudo-labels also can be biased towards a majority of data. Thus, the improvement from usage of the extra unlabeled data is limited. In our work, we focus on how to more correctly label the unlabeled data, which eventually provides better performance.

## 3 Preliminaries and Motivation

Given a simple binary classification from the data $P_{XY}$ with a mixture of two Gaussians, consider that each class data has the label $Y: +1\ or -1$. Also, consider the data distribution of $X|Y$ is $N\left(\mu_1, \sigma^2\right)$ when $Y = +1$. Similarly, when $Y = -1$, the data distribution of $X|Y$ is $N\left(\mu_2, \sigma^2\right)$, where $\mu_1 > \mu_2$. Given one sample x, if $x > \frac{\mu_1+\mu_2}{2}$, then x can be classified into $+1$; otherwise $-1$. Accordingly, the classifier can be expressed as $f(x) = \text{sign}\left(x - \frac{\mu_1+\mu_2}{2}\right)$, where the term $\frac{\mu_1+\mu_2}{2}$ needs to be learned based on the data set X and the corresponding label set Y.

However, given imbalanced training data, the term $\frac{\mu_1+\mu_2}{2}$ in the trained classifier will be shifted to the mean value of a minority class. If a majority of data has the label $Y = +1$, then the classifier can be derived as $f(x) = \text{sign}\left(x + \alpha - \frac{\mu_1+\mu_2}{2}\right)$, where $\alpha > 0$. Fig. 1a illustrates an example of a biased classifier, which focuses mainly on improving the classification performance of a majority class. Such a class imbalance issue can be resolved by balancing data class via data sampling approach such as over-sampling or under-sampling as shown in Fig. 1b: in this example, the predicted decision boundary

is closer to the actual boundary after using under-or over-sampling method. Similarly, sampling weighting methods also change the predicted decision boundary to the actual boundary.

Fig. 1c illustrates another example of a biased classifier, which focuses on improving the performance of a majority class. However, in this example, the number of data from a minority class is too small to generalize the data corresponding to the minority class. Since the data from the minority class does not generalize to the actual distribution, any sampling approach cannot improve the performance as shown in Fig. 1d: in this example, the predicted decision boundary is almost unchanged even after using under-or over-sampling method. Similarly, sampling weighting methods also have little effect on the predicted decision boundary.

To alleviate the class imbalance issue, Yang et al. [30] recently demonstrated that pseudo-label on extra unlabeled data can improve the classification performance, especially with the imbalanced dataset, theoretically and empirically. More specifically, a base classifier $f_B$ was first trained based on the original imbalanced training data. Subsequently, extra unlabeled data was labeled using $f_B$. At last, by re-training $f_B$ with the additional pseudo-label data, the classifier was shown to be improved. However, the pseudo-labels also can be biased towards a majority of data, which results in the incorrect labeling, especially for a minority of data. Thus, the improvement from usage of the extra unlabeled data is limited. In this study, we present the algorithms that can improve the labeling accuracy, which eventually improves the overall classification performance.

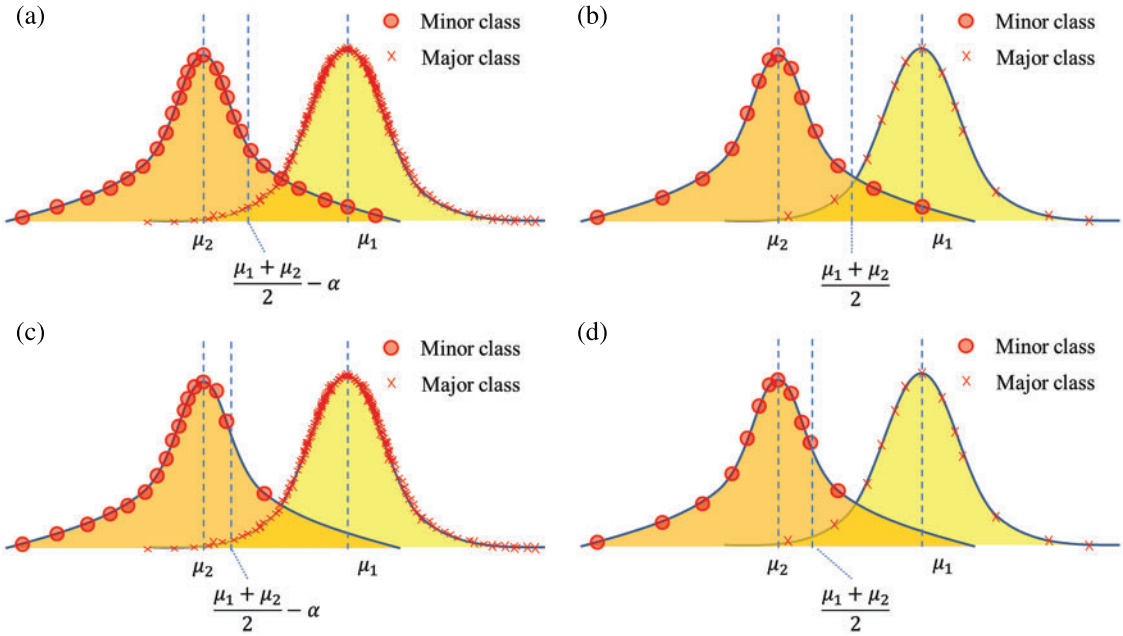


**Figure 1:** Examples of a biased classifier and the effects of data-level techniques; (a) an example of a biased classifier, (b) the effect of under-or over-sampling method (the predicted decision boundary closer to the actual boundary), (c) another example of a biased classifier, (d) the effect of under-or over-sampling method (little effect on the predicted decision boundary)

## 4 Iterative Semi-Supervised Learning Using Softmax Probability

### *4.1 Algorithm Description*

In this study, we propose the semi-supervised learning algorithms, which iteratively corrects the labeling of the extra unlabeled data. Algorithm 1 presents the pseudo-code of our proposed algorithm named iterative semi-supervised learning based on softmax probability (ISSL-SP). Let denote the original labeled data and the extra unlabeled data by $Data_{ori}$ and $Data_{un}$, respectively. Regarding the instance perspectives, let denote the $i^{th}$ extra unlabeled data and the corresponding label by $Data^i_{un}$ and $Label^i_{un}$, respectively. Let also denote the $i^{th}$ original labeled data and the corresponding label by $Data^i_{ori}$ and $Label^i_{ori}$, respectively. Before applying the algorithm ISSL-SP, we first train a base classifier $f_B$ using the original training data $Data_{ori}$. In the first stage, we consider the softmax probabilities corresponding to each class for $Data^i_{un}$, where $i = 1, 2, \ldots n\left(Data^i_{un}\right)$ for the number of unlabeled data. For each of $Data^i_{un}$, if the maximum value of the softmax probabilities is equal or greater than 0.99, we assigned the corresponding the class to $Label^i_{un}$. Here, the optimized threshold value of 0.99 was found throughout this study, and the trade-off between accuracy metrics and the threshold value is described in Results. On the other hand, if the maximum value of the softmax probabilities is less than 0.99, we assign the label $Label^i_{un}$ as undefined. Every iteration, we update $f_B$ using all available data for training: $f_B$ to $f_{new}$. Finally, we arrange the data with labels assigned as undefined, and repeat the entire process until all the data is labeled in a specific class. In this way, ISSL-SP improves the overall classification performance by assigning the labels only with high softmax probability.

**Algorithm 1** Iterative semi-supervised learning based on softmax probability (ISSL-SP). This algorithm is given a base classifier $f_B$ which was trained with original training data $Data_{ori}$. We consider that the data has the label: 1, 2, . . .

**Require**
1: $Data_{ori}$: original train data
2: $Data_{un}$ : extra unlabeled data
3: $f_B$ : base classifier providing softmax probability // $f_B$ was trained with $Data_{ori}$
4: **function** ISSL-SP ($f_B$, $Data_{un}$, $n\,(Data_{un})$) // $n\,(Data_{un})$ : the number of $Data_{un}$
5: $f_{new} = f_B$
6: **while** $n\,(Data_{un}) > 0$ **do**
7: **for** $i = 1$ to $n\,(Data_{un})$ **do**
8: // $Data^i_{un}$: $i^{th}$ *unlabeled data*
9: probs = $f_B\left(Data^i_{un}\right)$ // *softmax probabilities for each class*
10: **if** $max\,(probs) \geq 0.99$ **then**
11: // *0.99 or higher is considered correct*
12: $Label^i_{un} = argmax\,(probs)$
13: **else**
14: $Label^i_{un} = -1$ // *undefined*
15: **end if**
16: **end for**
17: Update $f_{new}$ based on the all available data including $Data_{ori}$ and $Data_{un}$ with $Label_{un} > 0$
18: Update $Data_{un}$ with $Label^i_{un} = -1$
19: **end while**
20: **return** $f_{new}$
21: **end function**

### 4.2 Algorithm Insight

Based on $Data^{i}_{un}$ with $Label^{i}_{un}$ from $f_B$, let denote the data corresponding to $Label^{i}_{un} = +1$ by $Data^{i+}_{un}$. Similarly, let denote the data corresponding to $Label^{i}_{un} = -1$ by $Data^{i-}_{un}$. As we mentioned above, our aim is to learn $\frac{\mu_1+\mu_2}{2}$. Here, with $Data^{i+}_{un}$ and $Data^{i-}_{un}$, the estimator can be constructed by

$$\theta = \frac{1}{2}\left(\sum_{i=1}^{n^+}\frac{Data^{i+}_{un}}{n^+} + \sum_{i=1}^{n^-}\frac{Data^{i-}_{un}}{n^-}\right), \tag{3}$$

where $n^+$ and $n^-$ are the numbers of the $Data^{i+}_{un}$ and $Data^{i-}_{un}$, respectively. Given the distribution of $Data^{i+}_{un} N\left(\mu_1, \sigma^2\right)$, and that of $Data^{i-}_{un} N\left(\mu_2, \sigma^2\right)$, the estimator can be expressed by

$$\begin{aligned}\theta &\sim \frac{1}{2}\left(\sum_{i=1}^{n^+}\frac{N\left(\mu_1, \sigma^2\right)}{n^+} + \sum_{i=1}^{n^-}\frac{N\left(\mu_1, \sigma^2\right)}{n^-}\right) \\ &\sim N\left(\mu_1 + \mu_2, \sigma^2\left(\frac{1}{n^+} + \frac{1}{n^-}\right)\right)\end{aligned} \tag{4}$$

The term $\sigma^2\left(\frac{1}{n^+} + \frac{1}{n^-}\right)$ decreases as $n^+$ and $n^-$ increase. Based on the assumption that the unlabeled data is correctly labelled, the estimation accuracy can increase as the number of unlabeled data increases. However, the base classifier $f_B$ based pseudo-labels can be biased towards a majority of data. Thus, we need to select only the pseudo-label data with high confidence and to train the model together with $Data_{ori}$. Then, the base classifier $f_B$ can be updated to the model providing higher accuracy, which labels the remained unlabeled data. By repeating the process over and over, the accuracy of the classifier model gradually improves.

### 4.3 A variant of ISSL-SP

ISSL-SP algorithm can be extended in a variety of forms. Algorithm 2 presents the pseudo-code named ISSL-SP with re-labeling all the initial unlabeled data (ISSL-SPR). As a variant of ISSL-SP, ISSL-SPR is the same as ISSL-SP, except that all of the unlabeled data is labeled again every iteration: the line 18 in ISSL-SP (Algorithm 1) is missing. Since the updated classifier $f_{new}$ is trained with ever increasing data, it can provide better performance as the process is repeated; and thus, it may be necessary for the initial unlabeled data $Data_{un}$ to be labeled over and over again. To sum up, ISSL-SP labels only the data assigned by undefined while ISSL-SPR labels all initial unlabeled data over again.

**Algorithm 2** A variant of ISSL-SP: ISSL-SPR. This algorithm is the same as ISSL-SP, except that all of the unlabeled data are labeled again.

**Require**

1: $Data_{ori}$ : original train data
2: $Data_{un}$ : extra unlabeled data
3: $f_B$ : base classifier providing softmax probability // $f_B$ was trained with $Data_{ori}$
4: **function** ISSL-SPR($f_B$, $Data_{un}$, $n\,(Data_{un})$) // $n(Data_{un})$ : the number of $Data_{un}$
5: $f_{new} = f_B$
6: **while** True **do**
7: ***Same from lines 7 to 17 in Algorithm 1***
8: **if** $n\,(Label_{un} == -1) == 0$ **then**

(Continued)

**Algorithm 2** Continued

9: break
10: **end if**
11: **end while**
12: **return** $f_{new}$
13: **end function**

## 5 Dataset and Experiment Setup

### *5.1 Dataset*

To evaluate our proposed algorithms of ISSL-SP and ISSL-SPR, we mainly used two datasets of CIFAR-10 [31] and the street view house number (SVHN) [32]. The two datasets include images and the corresponding class labels. In addition, they have additional unlabeled data with similar distributions: 80 Million Tiny Images [33] includes the unlabeled images for CIFAR-10, and extra SVHN [32] includes the unlabeled images for SVHN. Tab. 1 summarizes the four datasets of CIFAR-10, 80 Million Tiny Images, SVHN and extra SVHN. More specifically, for training, 80 Million Tiny Images includes 500,000 unlabeled images while CIFAR-10 includes 50,000 labeled images. The extra SVHN includes 531,131 unlabeled images while SVHN includes 73,257 images.

**Table 1:** Summary of four datasets: CIFAR-10, 80 Million Tiny Images, SVHN and extra SVHN. 80 Million Tiny Images are unlabeled images for CIFAR-10. Extra SVHN images are unlabeled images for SVHN

| CIFAR-10 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Class | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Total |
| $Data_{org}$ | 5,000 | 5,000 | 5,000 | 5,000 | 5,000 | 5,000 | 5,000 | 5,000 | 5,000 | 5,000 | 50,000 |
| $Data_{un}$ | 50,443 | 50,246 | 51,226 | 52,509 | 45,380 | 51,743 | 47,156 | 50,811 | 50,344 | 50,142 | 500,000 |
| **SVHN** | | | | | | | | | | | |
| Class | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Total |
| $Data_{org}$ | 4,948 | 13,861 | 10,585 | 8,497 | 7,458 | 6,882 | 5,727 | 5,595 | 5,045 | 4,659 | 73,257 |
| $Data_{un}$ | 45,550 | 90,560 | 74,740 | 60,765 | 50,633 | 53,490 | 41,582 | 43,997 | 35,358 | 34,456 | 531,131 |

### *5.2 Experimental Setup*

In this study, we conducted experiments on artificially created long-tailed data distribution from CIFAR-10 and SVHN. Tab. 2 summarizes the trained data randomly drawn from datasets of CIFAR-10, 80 Million Tiny Images, SVHN and extra SVHN. The class imbalance ratio was defined as the number of the most frequent class divided by that of the least frequent class [29–31].

**Table 2:** Summary of trained data randomly drawn from datasets from datasets of CIFAR-10, 80 Million Tiny Images, SVHN, extra SVHN and CINIC-10. For the unlabeled data $Data_{un}$, we considered two scenarios with different imbalance ratios

| CIFAR-10 | Imbalance ratio | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $Data_{org}$ | 50 | 5,000 | 3,237 | 2,096 | 1,357 | 878 | 568 | 368 | 238 | 154 | 100 | 13,996 |
| ***Scenario 1*** | | | | | | | | | | | | |
| $Data_{un}$ | 1 | 1,399 | 1,399 | 1,399 | 1,399 | 1,399 | 1,399 | 1,399 | 1,399 | 1,399 | 1,399 | 13,990 |
| ***Scenario 2*** | | | | | | | | | | | | |
| $Data_{un}$ | 50 | 5,005 | 3,237 | 2,096 | 1,355 | 876 | 568 | 368 | 236 | 153 | 96 | 13,990 |
| **SVHN** | Imbalance ratio | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Total |
| $Data_{org}$ | 50 | 20 | 1,000 | 647 | 419 | 271 | 175 | 113 | 73 | 47 | 30 | 2,795 |
| ***Scenario 1*** | | | | | | | | | | | | |
| $Data_{un}$ | 1 | 279 | 279 | 279 | 279 | 279 | 279 | 279 | 279 | 279 | 279 | 2,790 |
| ***Scenario 2*** | | | | | | | | | | | | |
| $Data_{un}$ | 50 | 18 | 1004 | 648 | 417 | 270 | 176 | 111 | 72 | 46 | 28 | 2,790 |
| **CINIC-10** | Imbalance ratio | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Total |
| $Data_{org}$ | 50 | 9,000 | 5,827 | 3,773 | 2,442 | 1,581 | 1,024 | 663 | 429 | 278 | 180 | 25,197 |
| ***Scenario 1*** | | | | | | | | | | | | |
| $Data_{un}$ | 1 | 2,519 | 2,519 | 2,519 | 2,519 | 2,519 | 2,519 | 2,519 | 2,519 | 2,519 | 2,519 | 25,190 |
| ***Scenario 2*** | | | | | | | | | | | | |
| $Data_{un}$ | 50 | 9,000 | 5,827 | 3,773 | 2,442 | 1,581 | 1,024 | 663 | 429 | 278 | 180 | 25,190 |

For CIFAR-10 and SVHN, we randomly drew samples to make the imbalance ratio of 50, which is denoted by $Data_{ori}$. For the unlabeled data $Data_{un}$, we considered two scenarios with different imbalance ratios. In Scenario 1, we assumed that the unlabeled data was balanced with the imbalance ratio of 1. In Scenario 2, we assumed that the unlabeled data was imbalanced with the imbalance ratio of 50. For both scenarios, we almost balanced out the numbers of labeled and unlabeled data: 13,996 $Data_{ori}$ and 13,990 $Data_{un}$ from CIFAR-10 and 80 Million Tiny Images while 2,795 $Data_{ori}$ and 2,790 $Data_{un}$ from SVHN and extra SVHN. Finally, we evaluated each of the trained models on the isolated and balanced testing dataset [30,31,34,35].

We implemented and trained the models using Pytorch. For all experiments, we used the stochastic gradient descent (SGD) optimizer with batch size of 256 and binary cross-entropy for the cost function. The entire experiments were performed on NVIDIA GeForce GTX 1080 Ti GPU.

### 5.3 Evaluation Metrics

To analyze the performance, the labeling percentage was defined as the number of the labeled data among $Data_{un}$ divided by the number of $Data_{un}$:

$$Labeled\ percentage = \frac{n(Data_{un}^{'})}{n(Data_{un})}, \tag{5}$$

where $Data_{un}^{'}$ is with the condition $Label_{un} > 0$ given $Data_{un}$.

To evaluate the performance, we used sensitivity (recall), specificity, precision, accuracy, balanced accuracy (BA) and F1 score as

$$Sensitivity = recall = \frac{TP}{TP + FN}, \tag{6}$$

$$precision = \frac{TP}{TP + FP}, \tag{7}$$

$$Specificity = \frac{TN}{TN + FP}, \tag{8}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{9}$$

$$Balanced\ Accuracy = \frac{Sensitivity + Specificity}{2}, \tag{10}$$

$$F1\ score = \frac{2 \times precision \times recall}{precision + recall}, \tag{11}$$

where TP, TN, FP, and FN represent the true positive, true negative, false positive, and false negative, respectively. In addition, we also used the metrics of top-1 error.

## 6 Results

### 6.1 With Balanced Unlabeled Data: Scenario 1

Tab. 3 summarizes the results when unlabeled data is balanced. It shows sensitivity, specificity, accuracy, BA, F1 score and top-1 error. Note since the testing dataset is balanced, the F1 score can be both macro average and weighted average. For the CIFAR-10 dataset, if only $Data_{ori}$ is used for training as a baseline, the top-1 error is 28.76%. If $Data_{un}$ is additionally used for training without iteration [30], the top-1 error is 24.93%, which is slightly decreased. On the other hand, if $Data_{un}$ is ideally given with 100% labeling accuracy and additionally used for training, the top-1 error is significantly dropped to 8.83%, which can be considered the lowest bound. Our proposed algorithms ISSL-SP and ISSL-SPR provide the top-1 error of 14.92% and 10.79%, respectively, which are much lower than that from the method [30], and are very close to the lowest bound. Similarly, for the SVHN dataset, with $Data_{ori}$ only, the top-1 error is 28.10%. If $Data_{un}$ is additionally used for training without iteration [30], the top-1 error decreases to 25.73%. With the ideal condition using $Data_{un}$ 100% labeling accuracy, the top-1 error is 9.17% as the lowest bound. Our proposed algorithms ISSL-SP and ISSL-SPR provide the top-1 error of 14.87% and 11.09%, respectively, which are also much lower than that from the method [30], and are very close to the lowest bound. More detailed results are presented in Supplementary

Tabs. 1 and 2. In addition, the results show that ISSL-SPR provides slightly higher accuracy than ISSL-SP, indicating that the updated classifier needs to re-label the entire initial unlabeled data.

**Table 3:** With balanced and unlabeled data from CIFAR-10 and SVHN datasets

| CIFAR-10 | Sensitivity | Specificity | Accuracy | BA | F1 | Top-1 |
|---|---|---|---|---|---|---|
| with $Data_{org}$ only | 0.7471 | 0.9719 | 0.9494 | 0.8595 | 0.7479 | 25.29 |
| without iteration [30], | 0.7696 | 0.9744 | 0.9539 | 0.8720 | 0.7718 | 23.04 |
| supervision learning | 0.9186 | 0.9910 | 0.9837 | 0.9548 | 0.9184 | 8.14 |
| (idally with 100% labeling accuracy) | | | | | | |
| ISSL-SP | 0.8543 | 0.9838 | 0.9709 | 0.9191 | 0.8555 | 14.57 |
| ISSL-SPR | 0.8955 | 0.9884 | 0.9791 | 0.9419 | 0.8959 | 10.45 |
| LDAM [31] | 0.8492 | 0.9832 | 0.9698 | 0.9162 | 0.8505 | 15.08 |
| PI [36] | 0.8525 | 0.9836 | 0.9705 | 0.9181 | 0.8522 | 14.75 |
| MT [37] | 0.8561 | 0.9840 | 0.9712 | 0.9201 | 0.8572 | 14.39 |
| VAT [16] | 0.8708 | 0.9856 | 0.9742 | 0.9282 | 0.8708 | 12.92 |
| ICT [38] | 0.8567 | 0.9841 | 0.9713 | 0.9204 | 0.8569 | 14.33 |
| FixMatch [39] | 0.8333 | 0.9870 | 0.9767 | 0.9352 | 0.8837 | 11.67 |
| **SVHN** | Sensitivity | Specificity | Accuracy | BA | F1 | Top-1 |
| with $Data_{org}$ only | 0.7822 | 0.9801 | 0.9654 | 0.8811 | 0.8044 | 17.33 |
| without iteration [30], | 0.8124 | 0.9820 | 0.9686 | 0.8972 | 0.8291 | 15.68 |
| supervision learning | 0.9406 | 0.9935 | 0.9884 | 0.9671 | 0.9398 | 5.78 |
| (idally with 100% labeling accuracy) | | | | | | |
| ISSL-SP | 0.9019 | 0.9903 | 0.9831 | 0.9461 | 0.9104 | 8.45 |
| ISSL-SPR | 0.9196 | 0.9919 | 0.9858 | 0.9558 | 0.9248 | 7.10 |
| LDAM [31] | 0.8513 | 0.9859 | 0.9754 | 0.9186 | 0.8650 | 12.31 |
| PI [36] | 0.9060 | 0.9903 | 0.9829 | 0.9481 | 0.9072 | 8.56 |
| MT [37] | 0.9110 | 0.9906 | 0.9835 | 0.9508 | 0.9140 | 8.23 |
| VAT [16] | 0.9013 | 0.9901 | 0.9826 | 0.9457 | 0.9070 | 8.37 |
| ICT [38] | 0.9153 | 0.9911 | 0.9842 | 0.9532 | 0.9145 | 7.90 |
| FixMatch [39] | 0.9046 | 0.9909 | 0.9841 | 0.9478 | 0.9143 | 7.95 |

Fig. 2 plots labeled percentages and top-1 errors using ISSL-SP and ISSL-SPR according to each iteration. It shows that the labeled percentage increases and top-1 error decreases as the labeling processing is repeated. Also, the tendency to change with each iteration can be observed in both algorithms of ISSL-SP and ISSL-SPR.

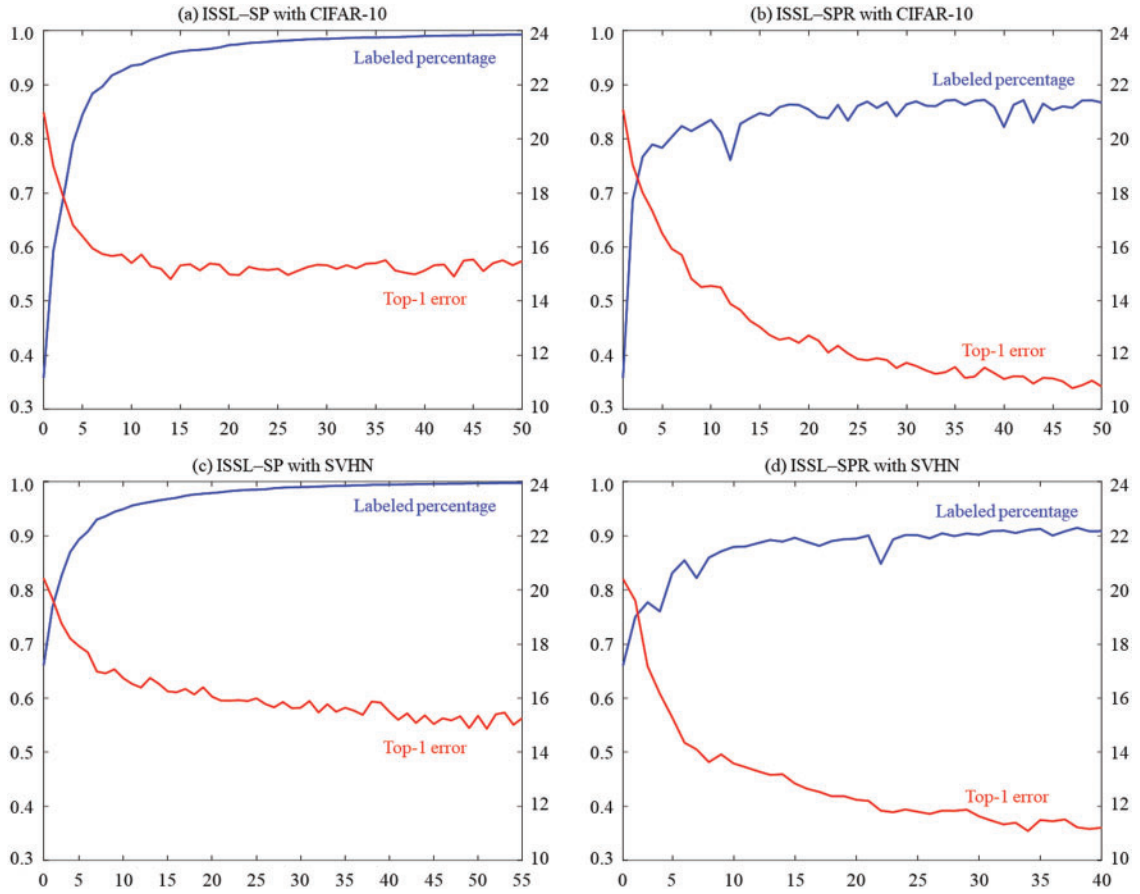


**Figure 2:** (Scenario 1: with balanced unlabeled data) Labeled percentages and top-1 errors using ISSL-SP and ISSL-SPR according to each iteration

### *6.2 With Balanced Unlabeled Data: Scenario 2*

Tab. 4 summarizes the results when unlabeled data is imbalanced. It shows sensitivity, specificity, accuracy, BA, F1 score and top-1 error. For the CIFAR-10 dataset, with $Data_{ori}$ only, the top-1 error is 28.76%. If $Data_{un}$ is additionally used for training without iteration [30], the top-1 error decreases to 25.85%. As the lowest bound, if $Data_{un}$ is ideally given with 100% labeling accuracy and additionally used for training, the top-1 error is 11.62%. Our proposed algorithms ISSL-SP and ISSL-SPR provide the top-1 error of 18.58% and 14.87%, respectively, which are also much lower than that from the method [30], and are very close to the lowest bound. Similarly, for the SVHN dataset, with $Data_{ori}$ only, the top-1 error is 28.10%. If $Data_{un}$ is additionally used for training without iteration [30], the top-1 error decreases to 25.25%. With the ideal condition using $Data_{un}$ 100% labeling accuracy, the top-1 error is 11.47% as the lowest bound. Our proposed algorithms ISSL-SP and ISSL-SPR provide the top-1 error of 14.14% and 13.62%, respectively, which are also much lower than that from the method [30], and are very close to the lowest bound. More detailed results are presented in Supplementary Tabs. 3 and 4. In addition, similar to the scenario 1, the results show that ISSL-SPR provides slightly

higher accuracy than ISSL-SP, indicating that the updated classifier needs to re-label the entire initial unlabeled data.

**Table 4:** With imbalanced and unlabeled data from CIFAR-10 and SVHN datasets

| CIFAR-10 | Sensitivity | Specificity | Accuracy | BA | F1 | Top-1 |
|---|---|---|---|---|---|---|
| with $Data_{org}$ only | 0.7471 | 0.9719 | 0.9494 | 0.8595 | 0.7479 | 25.29 |
| without iteration [30], | 0.7688 | 0.9743 | 0.8538 | 0.8716 | 0.7689 | 23.12 |
| supervision learning (idally with 100% labeling accuracy) | 0.8838 | 0.9871 | 0.9768 | 0.9354 | 0.8841 | 11.62 |
| ISSL-SP | 0.8383 | 0.9820 | 0.9677 | 0.9102 | 0.8393 | 16.17 |
| ISSL-SPR | 0.8723 | 0.9858 | 0.9745 | 0.9291 | 0.8731 | 12.77 |
| LDAM [31] | 0.8324 | 0.9814 | 0.9665 | 0.9069 | 0.8330 | 16.76 |
| PI [36] | 0.8037 | 0.9782 | 0.9607 | 0.8909 | 0.8054 | 19.63 |
| MT [37] | 0.8042 | 0.9782 | 0.9608 | 0.8912 | 0.8043 | 19.58 |
| VAT [16] | 0.8166 | 0.9796 | 0.9633 | 0.8981 | 0.8177 | 18.34 |
| ICT [38] | 0.7854 | 0.9762 | 0.9571 | 0.8808 | 0.7867 | 21.46 |
| FixMatch [39] | 0.8472 | 0.9830 | 0.9694 | 0.9151 | 0.8476 | 15.28 |
| **SVHN** | Sensitivity | Specificity | Accuracy | BA | F1 | Top-1 |
| with $Data_{org}$ only | 0.7822 | 0.9801 | 0.9654 | 0.8811 | 0.8044 | 17.33 |
| without iteration [30], | 0.7981 | 0.9816 | 0.9680 | 0.8898 | 0.8197 | 16.02 |
| supervision learning (idally with 100% labeling accuracy) | 0.8952 | 0.9904 | 0.9832 | 0.9428 | 0.9076 | 8.40 |
| ISSL-SP | 0.8437 | 0.9858 | 0.9755 | 0.9147 | 0.8648 | 12.23 |
| ISSL-SPR | 0.8761 | 0.9886 | 0.9801 | 0.9323 | 0.8886 | 9.96 |
| LDAM [31] | 0.8491 | 0.9850 | 0.9733 | 0.9170 | 0.8524 | 13.34 |
| PI [36] | 0.8665 | 0.9874 | 0.9780 | 0.9270 | 0.8771 | 11.20 |
| MT [37] | 0.8760 | 0.9887 | 0.9798 | 0.9323 | 0.8797 | 10.95 |
| VAT [16] | 0.8616 | 0.9927 | 0.9791 | 0.9245 | 0.8192 | 11.57 |
| ICT [38] | 0.8745 | 0.9886 | 0.9800 | 0.9316 | 0.8838 | 10.01 |
| FixMatch [39] | 0.8480 | 0.9864 | 0.9764 | 0.9172 | 0.8667 | 11.79 |

Fig. 3 plots labeled percentages and top-1 errors using ISSL-SP and ISSL-SPR according to each iteration. It also shows that the labeled percentage increases and top-1 error decreases as the labeling processing is repeated.

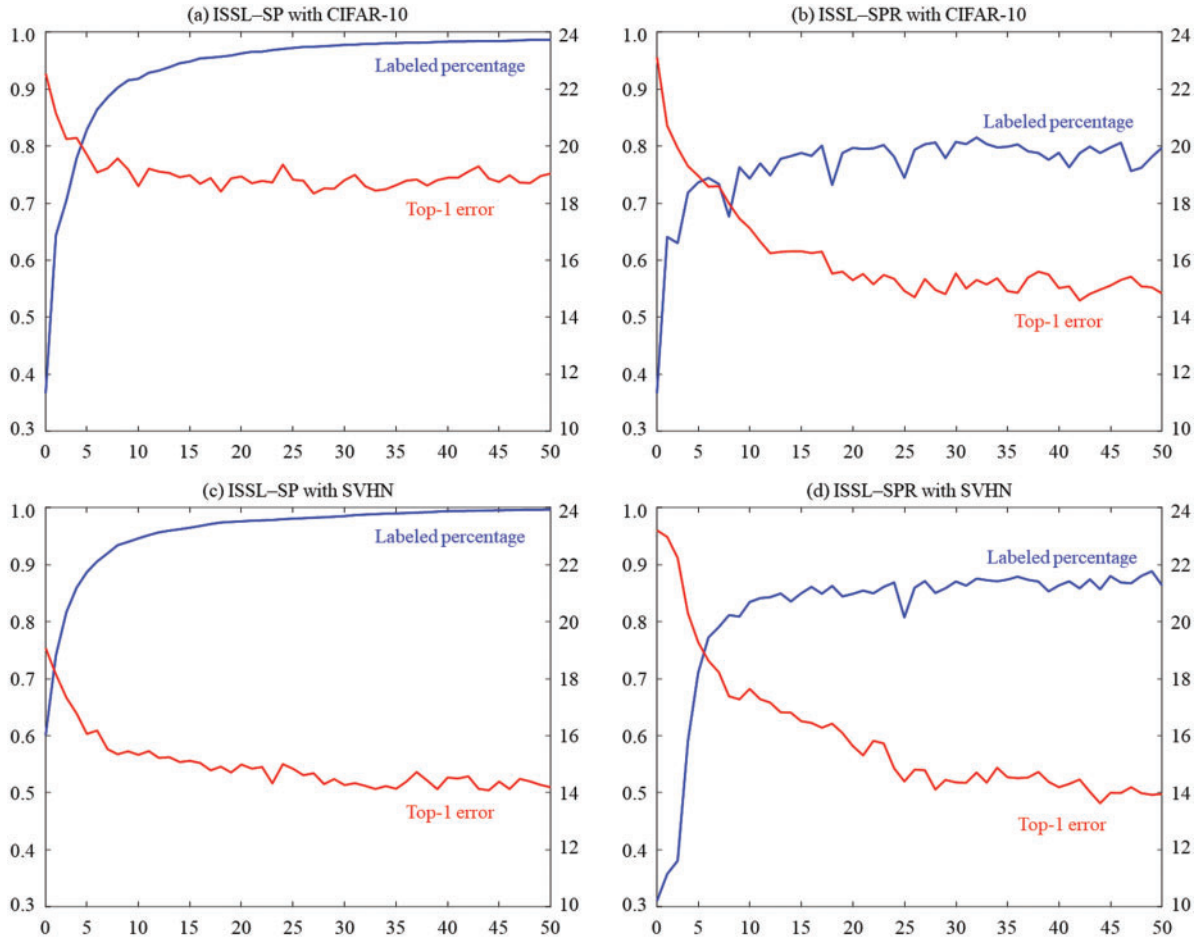


**Figure 3:** (Scenario 2: with balanced unlabeled data) Labeled percentages and top-1 errors using ISSL-SP and ISSL-SPR according to each iteration

### *6.3 Effect of Softmax Threshold Values*

To investigate the effect of the softmax threshold values, we changed the threshold values from 0.5 to 0.999: by the increment of 0.01 from 0.5 to 0.9, and the increment of 0.001 from 0.9 to 0.999. Fig. 4 shows the accuracy metrics of F1 score, balanced accuracy and top-1 error according to the softmax threshold values. The results show that the threshold value of 0.99 provides the highest accuracy values. Throughout this study, we have used the softmax threshold value of 0.99 for the simulation results.

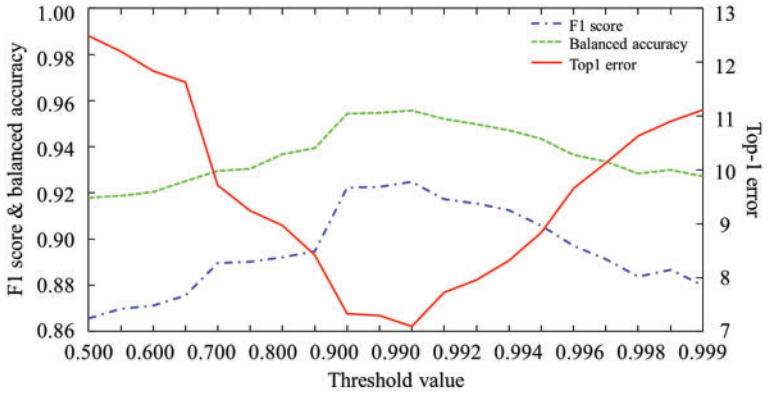


**Figure 4:** F1 score, Balanced accuracy and top-1 errors according to softmax threshold values

## 7 Conclusion and Discussion

In this study, we propose new semi-supervised learning algorithms, which iteratively corrects the labeling of the extra unlabeled data based on softmax probabilities. We first train a base classifier using original labeled data, and evaluate unlabeled data using softmax probabilities. For each unlabeled data, if the maximum value of the softmax probabilities is equal or greater than 0.99, we assign the unlabeled data with the corresponding class. Every iteration, we update the classifier using all available data for training. Regarding the labeling, ISSL-SP considers only the remaining unlabeled data while ISSL-SPR considers the entire initial unlabeled data. To validate the proposed algorithms, we tested on the two scenarios: with balanced unlabeled dataset and with imbalanced unlabeled dataset. The results show that the two proposed algorithms, ISSL-SP and ISSL-SPR, provide the accuracy as high as that from supervised learning, where the unlabeled data is given 100% labeling accuracy.

Comparing the performance of the two algorithms of ISSP-SP and ISSP-SPR, ISS-SPR outperforms ISS-SP regardless of the datasets and the imbalance ratio of unlabeled data. The results indicate that the updated classifier needs to re-label the entire initial unlabeled data. Furthermore, ISS-SPR outperforms previous state-of-the-arts. In the future work, we plan to validate the algorithm efficacy using more extended datasets. In addition, we need to investigate an optimum strategy to reduce the lengthy training time caused by the iteration process.

**Funding Statement:** This work was supported by the National Research Foundation of Korea (No. 2020R1A2C1014829), and by the Korea Medical Device Development Fund grant, which is funded by the Government of the Republic of Korea Korea government (the Ministry of Science and ICT; the Ministry of Trade, Industry and Energy; the Ministry of Health and Welfare; and the Ministry of Food and Drug Safety) (grant KMDF_PR_20200901_0095).

**Conflicts of Interest:** The authors declare that they have no conflicts of interest to report regarding the present study.

## References

[1] S. S. Nath, G. Mishra, J. Kar, S. Chakraborty and N. Dey, "A survey of image classification methods and techniques," in *2014 Int. Conf. on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, Kanyakumari District, India, pp. 554–557, 2014.

[2] E. Miranda, M. Aryuni and E. Irwansyah, "A survey of medical image classification techniques," in *2016 Int. Conf. on Information Management and Technology (ICIMTech)*, Bandung, Indonesia, pp. 56–61, 2016.
[3] E. Vocaturo, "Image classification techniques," in *Handbook of Research on Disease Prediction Through Data Analytics and Machine Learning*, Pennsylvania, USA, pp. 22–49. IGI Global, 2021.
[4] S. Shakya, "Analysis of artificial intelligence based image classification techniques," *Journal of Innovative Image Processing (JIIP)*, vol. 2, no. 1, pp. 44–54, 2020.
[5] W. Sun, G. Dai, X. Zhang, X. He and X. Chen, "TBE-Net: A three-branch embedding network with part-aware ability and feature complementary learning for vehicle re-identification," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2021. DOI 10.1109/TITS.2021.3130403.
[6] W. Sun, L. Dai, X. Zhang, P. Chang and X. He, "RSOD: Real-time small object detection algorithm in UAV-based traffic monitoring," *Applied Intelligence*, vol. 92, no. 6, pp. 1–16, 2021.
[7] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.
[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014. [Online]. Available: https://arxiv.org/abs/1409.1556.
[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed *et al.,* "Going deeper with convolutions," in *Proc. of the IEEE Conf. on computer vision and pattern recognition*, Boston, MA, USA, pp. 1–9, 2015.
[10] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proc. of the IEEE Conf. on computer vision and pattern recognition*, Las Vegas, NV, USA, pp. 770–778, 2016.
[11] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, pp. 2818–2826, 2016.
[12] J. M. Overhage and D. McCallie Jr, "Physician time spent using the electronic health record during outpatient encounters: A descriptive study," *Annals of Internal Medicine*, vol. 172, no. 3, pp. 169–174, 2020.
[13] M. J. Willemink, W. A. Koszek, C. Hardell, J. Wu, D. Fleischmann *et al.,* "Preparing medical imaging data for machine learning," *Radiology*, vol. 295, no. 1, pp. 4–15, 2020.
[14] Y. Xu, T. Mo, Q. Feng, P. Zhong, M. Lai *et al.,* "Deep learning of feature representation with multiple instance learning for medical image analysis," in *2014 IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Florence, Italy, pp. 1626–1630, 2014.
[15] J. E. Van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2020.
[16] T. Miyato, S.-i Maeda, M. Koyama and S. Ishii, "Virtual adversarial training: A regularization method for supervised and semi-supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.
[17] W. Bai, O. Oktay, M. Sinclair, H. Suzuki, M. Rajchl *et al.,* "Semi-supervised learning for network-based cardiac MR image segmentation," in *Int. Conf. on Medical Image Computing and Computer-Assisted Intervention*, Quebec City, Quebec, Canada, pp. 253–260, 2017.
[18] B. Krawczyk, "Learning from imbalanced data: Open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.
[19] H. Kaur, H. S. Pannu and A. K. Malhi, "A systematic review on imbalanced data challenges in machine learning: Applications and solutions," *ACM Computing Surveys*, vol. 52, no. 4, pp. 1–36, 2019.
[20] G. H. Nguyen, A. Bouzerdoum and S. L. Phung, "Learning pattern classification tasks with imbalanced data sets," *Pattern Recognition*, pp. 193–208, 2009. [Online]. Available: https://ro.uow.edu.au/infopapers/792/.
[21] Y. Yan, M. Chen, M.-L. Shyu and S.-C. Chen, "Deep learning for imbalanced multimedia data classification," in *2015 IEEE International Symposium on Multimedia (ISM)*, Miami, FL, USA, pp. 483–488, 2015.
[22] Y. Yan, M. Chen, M.-L. Shyu and S.-C. Chen, "Deep learning for imbalanced multimedia data classification," in *2015 IEEE International Symposium on Multimedia (ISM)*, Miami, FL, USA, pp. 483–488, 2015.

[23] N. V. Chawla, N. Japkowicz and A. Kotcz, "Special issue on learning from imbalanced data sets," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 1–6, 2004.
[24] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, pp. 1–54, 2019.
[25] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
[26] H. He, Y. Bai, E. A. Garcia and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE Int. Joint Conf. on Neural Networks (IEEE World Congress on Computational Intelligence)*, Hong Kong, China, pp. 1322–1328, 2008.
[27] C. Elkan, "The foundations of cost-sensitive learning," in *Int. Joint Conf. on Artificial Intelligence*, Seattle, Washington, USA, pp. 973–978, 2001.
[28] C. X. Ling and V. S. Sheng, "Cost-sensitive learning and the class imbalance problem," *Encyclopedia of Machine Learning*, vol. 2011, pp. 231–235, 2008.
[29] Y. Cui, M. Jia, T.-Y. Lin, Y. Song and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proc. of the IEEE/CVF Conf. on computer vision and pattern recognition*, Long Beach, CA, USA, pp. 9268–9277, 2019.
[30] Y. Yang and Z. Xu, "Rethinking the value of labels for improving class-imbalanced learning," 2020. [Online]. Available: https://arxiv.org/abs/2006.07529.
[31] K. Cao, C. Wei, A. Gaidon, N. Arechiga and T. Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," 2019. [Online]. Available: https://arxiv.org/abs/1906.07413.
[32] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu *et al.,* "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop*, Granada, Spain, 2011.
[33] A. Torralba, R. Fergus and W. T. Freeman, "80 Million tiny images: A large data set for nonparametric object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1958–1970, 2008.
[34] B. Kang, S. Xie, M. Rohrbach, Z. Yan, A. Gordo *et al.,* "Decoupling representation and classifier for long-tailed recognition," 2019. [Online]. Available: https://arxiv.org/abs/1910.09217.
[35] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong *et al.,* "Large-scale long-tailed recognition in an open world," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, pp. 2537–2546, 2019.
[36] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," 2016. [Online]. Available: https://arxiv.org/abs/1610.02242.
[37] A. Tarvainen and H. Valpola, "Weight-averaged consistency targets improve semi-supervised deep learning results," CoRR, 2017. [Online]. Available: https://arxiv.org/abs/1703.01780.
[38] V. Verma, K. Kawaguchi, A. Lamb, J. Kannala, Y. Bengio *et al.,* "Interpolation consistency training for semi-supervised learning," 2019. [Online]. Available: https://arxiv.org/abs/1903.03825.
[39] K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini *et al.,* "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," 2020. [Online]. Available: https://arxiv.org/abs/2001.07685.

**Supplementary Table 1:** Results from Scenario 1 with CIFAR-10

| with $Data_{org}$ only | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 8,203 | 797 | 73 | 927 | 0.9270 | 0.9114 | 0.9130 | 0.6806 | 0.9192 |
| 2 | 8,762 | 238 | 62 | 938 | 0.9380 | 0.9736 | 0.9700 | 0.8621 | 0.9558 |
| 3 | 8,725 | 275 | 316 | 684 | 0.6840 | 0.9694 | 0.9409 | 0.6983 | 0.8267 |
| 4 | 8,392 | 608 | 341 | 659 | 0.6590 | 0.9324 | 0.9051 | 0.5814 | 0.7957 |
| 5 | 8,669 | 331 | 255 | 745 | 0.7450 | 0.9632 | 0.9414 | 0.7177 | 0.8541 |
| 6 | 8,833 | 167 | 468 | 532 | 0.5320 | 0.9814 | 0.9365 | 0.6263 | 0.7567 |
| 7 | 8,853 | 147 | 263 | 737 | 0.7370 | 0.9837 | 0.9590 | 0.7824 | 0.8603 |
| 8 | 8,816 | 184 | 337 | 663 | 0.6630 | 0.9796 | 0.9479 | 0.7179 | 0.8213 |

(Continued)

**Supplementary Table 1:** Continued

| with $Data_{org}$ only | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
|---|---|---|---|---|---|---|---|---|---|
| 9 | 8,952 | 48 | 456 | 544 | 0.5440 | 0.9947 | 0.9496 | 0.6834 | 0.7693 |
| 10 | 8,919 | 81 | 305 | 695 | 0.6950 | 0.9910 | 0.9614 | 0.7827 | 0.8430 |
| mean | 8,712 | 288 | 288 | 712 | 0.7124 | 0.9680 | 0.9425 | 0.7133 | 0.8402 |
| without iteration | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 8,347 | 653 | 59 | 941 | 0.9410 | 0.9274 | 0.9288 | 0.7255 | 0.9342 |
| 2 | 8,778 | 222 | 42 | 958 | 0.9580 | 0.9753 | 0.9736 | 0.8789 | 0.9667 |
| 3 | 8,795 | 205 | 281 | 719 | 0.7190 | 0.9772 | 0.9514 | 0.7474 | 0.8481 |
| 4 | 8,261 | 739 | 219 | 781 | 0.7810 | 0.9179 | 0.9042 | 0.6198 | 0.8494 |
| 5 | 8,726 | 274 | 198 | 802 | 0.8020 | 0.9696 | 0.9528 | 0.7726 | 0.8858 |
| 6 | 8,891 | 109 | 430 | 570 | 0.5700 | 0.9879 | 0.9461 | 0.6790 | 0.7789 |
| 7 | 8,903 | 97 | 249 | 751 | 0.7510 | 0.9892 | 0.9654 | 0.8128 | 0.8701 |
| 8 | 8,882 | 118 | 306 | 694 | 0.6940 | 0.9869 | 0.9576 | 0.7660 | 0.8404 |
| 9 | 8,972 | 28 | 435 | 565 | 0.5650 | 0.9969 | 0.9537 | 0.7094 | 0.7809 |
| 10 | 8,952 | 48 | 274 | 726 | 0.7260 | 0.9947 | 0.9678 | 0.8185 | 0.8603 |
| mean | 8,751 | 249 | 249 | 751 | 0.7507 | 0.9723 | 0.9501 | 0.7530 | 0.8615 |
| supervision learning (idally with 100% labeling accuracy) | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 8,849 | 151 | 59 | 941 | 0.9410 | 0.9832 | 0.9790 | 0.8996 | 0.9621 |
| 2 | 8,956 | 44 | 28 | 972 | 0.9720 | 0.9951 | 0.9928 | 0.9643 | 0.9836 |
| 3 | 8,871 | 129 | 108 | 892 | 0.8920 | 0.9857 | 0.9763 | 0.8827 | 0.9388 |
| 4 | 8,836 | 164 | 185 | 815 | 0.8150 | 0.9818 | 0.9651 | 0.8236 | 0.8984 |
| 5 | 8,895 | 105 | 73 | 927 | 0.9270 | 0.9883 | 0.9822 | 0.9124 | 0.9577 |
| 6 | 8,891 | 109 | 161 | 839 | 0.8390 | 0.9879 | 0.9730 | 0.8614 | 0.9134 |
| 7 | 8,925 | 75 | 50 | 950 | 0.9500 | 0.9917 | 0.9875 | 0.9383 | 0.9708 |
| 8 | 8,962 | 38 | 82 | 918 | 0.9180 | 0.9958 | 0.9880 | 0.9387 | 0.9569 |
| 9 | 8,969 | 31 | 69 | 931 | 0.9310 | 0.9966 | 0.9900 | 0.9490 | 0.9638 |
| 10 | 8,963 | 37 | 68 | 932 | 0.9320 | 0.9959 | 0.9895 | 0.9467 | 0.9639 |
| mean | 8,912 | 88 | 88 | 912 | 0.9117 | 0.9902 | 0.9823 | 0.9117 | 0.9509 |
| ISSL-SP | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 8,659 | 341 | 48 | 952 | 0.9520 | 0.9621 | 0.9611 | 0.8304 | 0.9571 |
| 2 | 8,866 | 134 | 20 | 980 | 0.9800 | 0.9851 | 0.9846 | 0.9272 | 0.9826 |
| 3 | 8,794 | 206 | 175 | 825 | 0.8250 | 0.9771 | 0.9619 | 0.8124 | 0.9011 |
| 4 | 8,699 | 301 | 217 | 783 | 0.7830 | 0.9666 | 0.9482 | 0.7514 | 0.8748 |
| 5 | 8,842 | 158 | 139 | 861 | 0.8610 | 0.9824 | 0.9703 | 0.8529 | 0.9217 |
| 6 | 8,865 | 135 | 205 | 795 | 0.7950 | 0.9850 | 0.9660 | 0.8238 | 0.8900 |
| 7 | 8,898 | 102 | 138 | 862 | 0.8620 | 0.9887 | 0.9760 | 0.8778 | 0.9253 |
| 8 | 8,947 | 53 | 174 | 826 | 0.8260 | 0.9941 | 0.9773 | 0.8792 | 0.9101 |
| 9 | 8,977 | 23 | 227 | 773 | 0.7730 | 0.9974 | 0.9750 | 0.8608 | 0.8852 |
| 10 | 8,961 | 39 | 149 | 851 | 0.8510 | 0.9957 | 0.9812 | 0.9005 | 0.9233 |
| mean | 8,851 | 149 | 149 | 692 | 0.8508 | 0.9834 | 0.9702 | 0.8516 | 0.9171 |
| ISSL-SPR | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 8,802 | 198 | 39 | 961 | 0.9610 | 0.9780 | 0.9763 | 0.8902 | 0.9695 |
| 2 | 8,948 | 52 | 27 | 973 | 0.9730 | 0.9942 | 0.9921 | 0.9610 | 0.9836 |
| 3 | 8,836 | 164 | 118 | 882 | 0.8820 | 0.9818 | 0.9718 | 0.8622 | 0.9319 |
| 4 | 8,789 | 211 | 190 | 810 | 0.8100 | 0.9766 | 0.9599 | 0.8016 | 0.8933 |
| 5 | 8,875 | 125 | 116 | 884 | 0.8840 | 0.9861 | 0.9759 | 0.8800 | 0.9351 |
| 6 | 8,854 | 146 | 169 | 831 | 0.8310 | 0.9838 | 0.9685 | 0.8407 | 0.9074 |
| 7 | 8,899 | 101 | 87 | 913 | 0.9130 | 0.9888 | 0.9812 | 0.9067 | 0.9509 |

(Continued)

**Supplementary Table 1:** Continued

| with $Data_{org}$ only | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
|---|---|---|---|---|---|---|---|---|---|
| 8 | 8,968 | 32 | 123 | 877 | 0.8770 | 0.9964 | 0.9845 | 0.9188 | 0.9367 |
| 9 | 8,981 | 19 | 129 | 871 | 0.8710 | 0.9979 | 0.9852 | 0.9217 | 0.9344 |
| 10 | 8,969 | 31 | 81 | 919 | 0.9190 | 0.9966 | 0.9888 | 0.9426 | 0.9578 |
| mean | 8,892 | 108 | 108 | 692 | 0.8921 | 0.9880 | 0.9784 | 0.8925 | 0.9401 |

**Supplementary Table 2:** Results from Scenario 1 with SVHN

| with $Data_{org}$ only | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 24,131 | 157 | 879 | 865 | 0.4960 | 0.9935 | 0.9602 | 0.6255 | 0.7448 |
| 2 | 19,897 | 1,036 | 376 | 4,723 | 0.9263 | 0.9505 | 0.9458 | 0.8700 | 0.9384 |
| 3 | 19,578 | 2,305 | 437 | 3,712 | 0.8947 | 0.8947 | 0.8947 | 0.7303 | 0.8947 |
| 4 | 21,984 | 1,166 | 1,072 | 1,810 | 0.6280 | 0.9496 | 0.9140 | 0.6180 | 0.7888 |
| 5 | 23,190 | 319 | 500 | 2,023 | 0.8018 | 0.9864 | 0.9685 | 0.8317 | 0.8941 |
| 6 | 22,662 | 986 | 498 | 1,886 | 0.7911 | 0.9583 | 0.9430 | 0.7177 | 0.8747 |
| 7 | 23,466 | 589 | 902 | 1,075 | 0.5438 | 0.9755 | 0.9427 | 0.5905 | 0.7596 |
| 8 | 23,887 | 126 | 695 | 1,324 | 0.6558 | 0.9948 | 0.9685 | 0.7633 | 0.8253 |
| 9 | 24,058 | 314 | 926 | 734 | 0.4422 | 0.9871 | 0.9524 | 0.5421 | 0.7146 |
| 10 | 24,121 | 316 | 1,029 | 566 | 0.3549 | 0.9871 | 0.9483 | 0.4570 | 0.6710 |
| mean | 22,697 | 731 | 731 | 1,872 | 0.6534 | 0.9678 | 0.9438 | 0.6746 | 0.8106 |
| without iteration | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 24,165 | 123 | 793 | 951 | 0.5453 | 0.9949 | 0.9648 | 0.6749 | 0.7701 |
| 2 | 19,694 | 1,239 | 287 | 4,812 | 0.9437 | 0.9408 | 0.9414 | 0.8631 | 0.9423 |
| 3 | 19,815 | 2,068 | 365 | 3,784 | 0.9120 | 0.9055 | 0.9065 | 0.7567 | 0.9088 |
| 4 | 22,164 | 986 | 955 | 1,927 | 0.6686 | 0.9574 | 0.9254 | 0.6651 | 0.8130 |
| 5 | 23,163 | 346 | 409 | 2,114 | 0.8379 | 0.9853 | 0.9710 | 0.8485 | 0.9116 |
| 6 | 22,878 | 770 | 479 | 1,905 | 0.7991 | 0.9674 | 0.9520 | 0.7531 | 0.8833 |
| 7 | 23,520 | 535 | 831 | 1,146 | 0.5797 | 0.9778 | 0.9475 | 0.6266 | 0.7787 |
| 8 | 23,927 | 86 | 694 | 1,325 | 0.6563 | 0.9964 | 0.9700 | 0.7726 | 0.8263 |
| 9 | 24,048 | 324 | 862 | 798 | 0.4807 | 0.9867 | 0.9544 | 0.5737 | 0.7337 |
| 10 | 24,210 | 227 | 1,029 | 566 | 0.3549 | 0.9907 | 0.9518 | 0.4740 | 0.6728 |
| mean | 22,758 | 670 | 670 | 1,933 | 0.6778 | 0.9703 | 0.9485 | 0.7008 | 0.8241 |
| supervision learning (idally with 100% labeling accuracy) | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 24,119 | 169 | 120 | 1,624 | 0.9312 | 0.9930 | 0.9889 | 0.9183 | 0.9621 |
| 2 | 20,574 | 359 | 284 | 4,815 | 0.9443 | 0.9829 | 0.9753 | 0.9374 | 0.9636 |
| 3 | 21,603 | 280 | 315 | 3,834 | 0.9241 | 0.9872 | 0.9771 | 0.9280 | 0.9556 |
| 4 | 22,684 | 466 | 340 | 2,542 | 0.8820 | 0.9799 | 0.9690 | 0.8632 | 0.9309 |
| 5 | 23,340 | 169 | 163 | 2,360 | 0.9354 | 0.9928 | 0.9872 | 0.9343 | 0.9641 |
| 6 | 23,433 | 215 | 262 | 2,122 | 0.8901 | 0.9909 | 0.9817 | 0.8990 | 0.9405 |
| 7 | 23,805 | 250 | 238 | 1,739 | 0.8796 | 0.9896 | 0.9813 | 0.8770 | 0.9346 |
| 8 | 23,902 | 111 | 244 | 1,775 | 0.8791 | 0.9954 | 0.9864 | 0.9091 | 0.9373 |
| 9 | 24,244 | 128 | 264 | 1,396 | 0.8410 | 0.9947 | 0.9849 | 0.8769 | 0.9179 |
| 10 | 24,197 | 240 | 157 | 1,438 | 0.9016 | 0.9902 | 0.9847 | 0.8787 | 0.9459 |
| mean | 23,190 | 239 | 239 | 2,365 | 0.9008 | 0.9897 | 0.9817 | 0.9022 | 0.9452 |

(Continued)

**Supplementary Table 2:** Continued

| with $Data_{org}$ only | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
|---|---|---|---|---|---|---|---|---|---|
| ISSL-SP | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 24,120 | 168 | 458 | 1,286 | 0.7374 | 0.9931 | 0.9760 | 0.8043 | 0.8652 |
| 2 | 20,218 | 715 | 280 | 4,819 | 0.9451 | 0.9658 | 0.9618 | 0.9064 | 0.9555 |
| 3 | 21,110 | 773 | 361 | 3,788 | 0.9130 | 0.9647 | 0.9564 | 0.8698 | 0.9388 |
| 4 | 22,326 | 824 | 413 | 2,469 | 0.8567 | 0.9644 | 0.9525 | 0.7997 | 0.9106 |
| 5 | 23,272 | 237 | 260 | 2,263 | 0.8969 | 0.9899 | 0.9809 | 0.9011 | 0.9434 |
| 6 | 23,315 | 333 | 326 | 2,058 | 0.8633 | 0.9859 | 0.9747 | 0.8620 | 0.9246 |
| 7 | 23,694 | 361 | 485 | 1,492 | 0.7547 | 0.9850 | 0.9675 | 0.7791 | 0.8698 |
| 8 | 23,870 | 143 | 340 | 1,679 | 0.8316 | 0.9940 | 0.9814 | 0.8743 | 0.9128 |
| 9 | 24,174 | 198 | 415 | 1,245 | 0.7500 | 0.9919 | 0.9765 | 0.8024 | 0.8709 |
| 10 | 24,319 | 118 | 532 | 1,063 | 0.6665 | 0.9952 | 0.9750 | 0.7659 | 0.8308 |
| mean | 23,042 | 387 | 387 | 692 | 0.8215 | 0.9830 | 0.9703 | 0.8365 | 0.9023 |
| ISSL-SPR | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 24,199 | 89 | 294 | 1,450 | 0.8314 | 0.9963 | 0.9853 | 0.8833 | 0.9139 |
| 2 | 20,254 | 679 | 208 | 4,891 | 0.9592 | 0.9676 | 0.9659 | 0.9169 | 0.9634 |
| 3 | 21,390 | 493 | 271 | 3,878 | 0.9347 | 0.9775 | 0.9707 | 0.9103 | 0.9561 |
| 4 | 22,687 | 463 | 384 | 2,498 | 0.8668 | 0.9800 | 0.9675 | 0.8550 | 0.9234 |
| 5 | 23,314 | 195 | 208 | 2,315 | 0.9176 | 0.9917 | 0.9845 | 0.9199 | 0.9546 |
| 6 | 23,433 | 215 | 286 | 2,098 | 0.8800 | 0.9909 | 0.9808 | 0.8933 | 0.9355 |
| 7 | 23,649 | 406 | 226 | 1,751 | 0.8857 | 0.9831 | 0.9757 | 0.8471 | 0.9344 |
| 8 | 23,918 | 95 | 293 | 1,726 | 0.8549 | 0.9960 | 0.9851 | 0.8990 | 0.9255 |
| 9 | 24,254 | 118 | 406 | 1,254 | 0.7554 | 0.9952 | 0.9799 | 0.8272 | 0.8753 |
| 10 | 24,303 | 134 | 311 | 1,284 | 0.8050 | 0.9945 | 0.9829 | 0.8523 | 0.8998 |
| mean | 23,140 | 289 | 289 | 692 | 0.8691 | 0.9873 | 0.9778 | 0.8804 | 0.9282 |

**Supplementary Table 3:** Results from Scenario 2 with CIFAR-10

| with $Data_{org}$ only | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 8,203 | 797 | 73 | 927 | 0.9270 | 0.9114 | 0.9130 | 0.6806 | 0.9192 |
| 2 | 8,762 | 238 | 62 | 938 | 0.9380 | 0.9736 | 0.9700 | 0.8621 | 0.9558 |
| 3 | 8,725 | 275 | 316 | 684 | 0.6840 | 0.9694 | 0.9409 | 0.6983 | 0.8267 |
| 4 | 8,392 | 608 | 341 | 659 | 0.6590 | 0.9324 | 0.9051 | 0.5814 | 0.7957 |
| 5 | 8,669 | 331 | 255 | 745 | 0.7450 | 0.9632 | 0.9414 | 0.7177 | 0.8541 |
| 6 | 8,833 | 167 | 468 | 532 | 0.5320 | 0.9814 | 0.9365 | 0.6263 | 0.7567 |
| 7 | 8,853 | 147 | 263 | 737 | 0.7370 | 0.9837 | 0.9590 | 0.7824 | 0.8603 |
| 8 | 8,816 | 184 | 337 | 663 | 0.6630 | 0.9796 | 0.9479 | 0.7179 | 0.8213 |
| 9 | 8,952 | 48 | 456 | 544 | 0.5440 | 0.9947 | 0.9496 | 0.6834 | 0.7693 |
| 10 | 8,919 | 81 | 305 | 695 | 0.6950 | 0.9910 | 0.9614 | 0.7827 | 0.8430 |
| mean | 8,712 | 288 | 288 | 712 | 0.7124 | 0.9680 | 0.9425 | 0.7133 | 0.8402 |
| without iteration | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 8,450 | 550 | 71 | 929 | 0.9290 | 0.9389 | 0.9379 | 0.7495 | 0.9339 |
| 2 | 8,770 | 230 | 40 | 960 | 0.9600 | 0.9744 | 0.9730 | 0.8767 | 0.9672 |
| 3 | 8,736 | 264 | 286 | 714 | 0.7140 | 0.9707 | 0.9450 | 0.7219 | 0.8423 |
| 4 | 8,335 | 665 | 284 | 716 | 0.7160 | 0.9261 | 0.9051 | 0.6014 | 0.8211 |
| 5 | 8,592 | 408 | 185 | 815 | 0.8150 | 0.9547 | 0.9407 | 0.7332 | 0.8848 |
| 6 | 8,851 | 149 | 425 | 575 | 0.5750 | 0.9834 | 0.9426 | 0.6671 | 0.7792 |

(Continued)

**Supplementary Table 3:** Continued

| with $Data_{org}$ only | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
|---|---|---|---|---|---|---|---|---|---|
| 7 | 8,921 | 79 | 315 | 685 | 0.6850 | 0.9912 | 0.9606 | 0.7766 | 0.8381 |
| 8 | 8,872 | 128 | 335 | 665 | 0.6650 | 0.9858 | 0.9537 | 0.7418 | 0.8254 |
| 9 | 8,930 | 70 | 310 | 690 | 0.6900 | 0.9922 | 0.9620 | 0.7841 | 0.8411 |
| 10 | 8,958 | 42 | 334 | 666 | 0.6660 | 0.9953 | 0.9624 | 0.7799 | 0.8307 |
| mean | 8,742 | 259 | 259 | 742 | 0.7415 | 0.9713 | 0.9483 | 0.7432 | 0.8564 |
| supervision learning (idally with 100% labeling accuracy) | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 8,851 | 149 | 63 | 937 | 0.9370 | 0.9834 | 0.9788 | 0.8984 | 0.9602 |
| 2 | 8,864 | 136 | 14 | 986 | 0.9860 | 0.9849 | 0.9850 | 0.9293 | 0.9854 |
| 3 | 8,843 | 157 | 113 | 887 | 0.8870 | 0.9826 | 0.9730 | 0.8679 | 0.9348 |
| 4 | 8,760 | 240 | 170 | 830 | 0.8300 | 0.9733 | 0.9590 | 0.8019 | 0.9017 |
| 5 | 8,829 | 171 | 69 | 931 | 0.9310 | 0.9810 | 0.9760 | 0.8858 | 0.9560 |
| 6 | 8,860 | 140 | 178 | 822 | 0.8220 | 0.9844 | 0.9682 | 0.8379 | 0.9032 |
| 7 | 8,906 | 94 | 108 | 892 | 0.8920 | 0.9896 | 0.9798 | 0.8983 | 0.9408 |
| 8 | 8,969 | 31 | 147 | 853 | 0.8530 | 0.9966 | 0.9822 | 0.9055 | 0.9248 |
| 9 | 8,970 | 30 | 134 | 866 | 0.8660 | 0.9967 | 0.9836 | 0.9135 | 0.9313 |
| 10 | 8,986 | 14 | 166 | 834 | 0.8340 | 0.9984 | 0.9820 | 0.9026 | 0.9162 |
| mean | 8,884 | 116 | 116 | 884 | 0.8838 | 0.9871 | 0.9768 | 0.8841 | 0.9354 |
| ISSL-SP | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 8,517 | 483 | 41 | 959 | 0.9590 | 0.9463 | 0.9476 | 0.7854 | 0.9527 |
| 2 | 8,803 | 197 | 22 | 978 | 0.9780 | 0.9781 | 0.9781 | 0.8993 | 0.9781 |
| 3 | 8,740 | 260 | 145 | 855 | 0.8550 | 0.9711 | 0.9595 | 0.8085 | 0.9131 |
| 4 | 8,640 | 360 | 230 | 770 | 0.7700 | 0.9600 | 0.9410 | 0.7230 | 0.8650 |
| 5 | 8,764 | 236 | 125 | 875 | 0.8750 | 0.9738 | 0.9639 | 0.8290 | 0.9244 |
| 6 | 8,838 | 162 | 262 | 738 | 0.7380 | 0.9820 | 0.9576 | 0.7768 | 0.8600 |
| 7 | 8,929 | 71 | 212 | 788 | 0.7880 | 0.9921 | 0.9717 | 0.8478 | 0.8901 |
| 8 | 8,958 | 42 | 266 | 734 | 0.7340 | 0.9953 | 0.9692 | 0.8266 | 0.8647 |
| 9 | 8,975 | 25 | 304 | 696 | 0.6960 | 0.9972 | 0.9671 | 0.8088 | 0.8466 |
| 10 | 8,978 | 22 | 251 | 749 | 0.7490 | 0.9976 | 0.9727 | 0.8458 | 0.8733 |
| mean | 8,814 | 186 | 186 | 692 | 0.8142 | 0.9794 | 0.9628 | 0.8151 | 0.8968 |
| ISSL-SPR | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 8,619 | 381 | 24 | 976 | 0.9760 | 0.9577 | 0.9595 | 0.8282 | 0.9668 |
| 2 | 8,855 | 145 | 16 | 984 | 0.9840 | 0.9839 | 0.9839 | 0.9244 | 0.9839 |
| 3 | 8,814 | 186 | 112 | 888 | 0.8880 | 0.9793 | 0.9702 | 0.8563 | 0.9337 |
| 4 | 8,668 | 332 | 185 | 815 | 0.8150 | 0.9631 | 0.9483 | 0.7592 | 0.8891 |
| 5 | 8,811 | 189 | 104 | 896 | 0.8960 | 0.9790 | 0.9707 | 0.8595 | 0.9375 |
| 6 | 8,896 | 104 | 248 | 752 | 0.7520 | 0.9884 | 0.9648 | 0.8103 | 0.8702 |
| 7 | 8,911 | 89 | 150 | 850 | 0.8500 | 0.9901 | 0.9761 | 0.8767 | 0.9201 |
| 8 | 8,973 | 27 | 220 | 780 | 0.7800 | 0.9970 | 0.9753 | 0.8633 | 0.8885 |
| 9 | 8,986 | 14 | 246 | 754 | 0.7540 | 0.9984 | 0.9740 | 0.8529 | 0.8762 |
| 10 | 8,980 | 20 | 182 | 818 | 0.8180 | 0.9978 | 0.9798 | 0.8901 | 0.9079 |
| mean | 8,851 | 149 | 149 | 692 | 0.8513 | 0.9835 | 0.9703 | 0.8521 | 0.9174 |

**Supplementary Table 4:** Results from Scenario 2 with SVHN

| with $Data_{org}$ only | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 24,131 | 157 | 879 | 865 | 0.4960 | 0.9935 | 0.9602 | 0.6255 | 0.7448 |
| 2 | 19,897 | 1,036 | 376 | 4,723 | 0.9263 | 0.9505 | 0.9458 | 0.8700 | 0.9384 |
| 3 | 19,578 | 2,305 | 437 | 3,712 | 0.8947 | 0.8947 | 0.8947 | 0.7303 | 0.8947 |
| 4 | 21,984 | 1,166 | 1,072 | 1,810 | 0.6280 | 0.9496 | 0.9140 | 0.6180 | 0.7888 |
| 5 | 23,190 | 319 | 500 | 2,023 | 0.8018 | 0.9864 | 0.9685 | 0.8317 | 0.8941 |
| 6 | 22,662 | 986 | 498 | 1,886 | 0.7911 | 0.9583 | 0.9430 | 0.7177 | 0.8747 |
| 7 | 23,466 | 589 | 902 | 1,075 | 0.5438 | 0.9755 | 0.9427 | 0.5905 | 0.7596 |
| 8 | 23,887 | 126 | 695 | 1,324 | 0.6558 | 0.9948 | 0.9685 | 0.7633 | 0.8253 |
| 9 | 24,058 | 314 | 926 | 734 | 0.4422 | 0.9871 | 0.9524 | 0.5421 | 0.7146 |
| 10 | 24,121 | 316 | 1,029 | 566 | 0.3549 | 0.9871 | 0.9483 | 0.4570 | 0.6710 |
| mean | 22,697 | 731 | 731 | 1,872 | 0.6534 | 0.9678 | 0.9438 | 0.6746 | 0.8106 |
| without iteration | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 24,105 | 183 | 742 | 1,002 | 0.5745 | 0.9925 | 0.9645 | 0.6842 | 0.7835 |
| 2 | 19,952 | 981 | 347 | 4,752 | 0.9319 | 0.9531 | 0.9490 | 0.8774 | 0.9425 |
| 3 | 20,098 | 1,785 | 451 | 3,698 | 0.8913 | 0.9184 | 0.9141 | 0.7679 | 0.9049 |
| 4 | 22,488 | 662 | 1,148 | 1,734 | 0.6017 | 0.9714 | 0.9305 | 0.6571 | 0.7865 |
| 5 | 23,262 | 247 | 551 | 1,972 | 0.7816 | 0.9895 | 0.9693 | 0.8317 | 0.8856 |
| 6 | 22,718 | 930 | 386 | 1,998 | 0.8381 | 0.9607 | 0.9494 | 0.7523 | 0.8994 |
| 7 | 23,109 | 946 | 550 | 1,427 | 0.7218 | 0.9607 | 0.9425 | 0.6561 | 0.8412 |
| 8 | 23,944 | 69 | 103 | 1,316 | 0.9274 | 0.9971 | 0.9932 | 0.9387 | 0.9623 |
| 9 | 24,090 | 282 | 928 | 732 | 0.4410 | 0.9884 | 0.9535 | 0.5475 | 0.7147 |
| 10 | 23,953 | 484 | 763 | 832 | 0.5216 | 0.9802 | 0.9521 | 0.5716 | 0.7509 |
| mean | 22,772 | 657 | 597 | 1,946 | 0.7231 | 0.9712 | 0.9518 | 0.7284 | 0.8471 |
| supervision learning (idally with 100% labeling accuracy) | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 24,227 | 61 | 378 | 1,366 | 0.7833 | 0.9975 | 0.9831 | 0.8616 | 0.8904 |
| 2 | 20,356 | 577 | 165 | 4,934 | 0.9676 | 0.9724 | 0.9715 | 0.9301 | 0.9700 |
| 3 | 21,202 | 681 | 195 | 3,954 | 0.9530 | 0.9689 | 0.9663 | 0.9003 | 0.9609 |
| 4 | 22,592 | 558 | 323 | 2,559 | 0.8879 | 0.9759 | 0.9662 | 0.8531 | 0.9319 |
| 5 | 23,282 | 227 | 167 | 2,356 | 0.9338 | 0.9903 | 0.9849 | 0.9228 | 0.9621 |
| 6 | 23,478 | 170 | 292 | 2,092 | 0.8775 | 0.9928 | 0.9823 | 0.9006 | 0.9352 |
| 7 | 23,615 | 440 | 248 | 1,729 | 0.8746 | 0.9817 | 0.9736 | 0.8341 | 0.9281 |
| 8 | 23,969 | 44 | 370 | 1,649 | 0.8167 | 0.9982 | 0.9841 | 0.8885 | 0.9075 |
| 9 | 24,239 | 133 | 440 | 1,220 | 0.7349 | 0.9945 | 0.9780 | 0.8098 | 0.8647 |
| 10 | 24,342 | 95 | 408 | 1,187 | 0.7442 | 0.9961 | 0.9807 | 0.8252 | 0.8702 |
| mean | 23,130 | 299 | 299 | 2,305 | 0.8574 | 0.9868 | 0.9771 | 0.8726 | 0.9221 |
| ISSL-SP | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 24,263 | 25 | 680 | 1,064 | 0.6101 | 0.9990 | 0.9729 | 0.7511 | 0.8045 |
| 2 | 20,257 | 676 | 188 | 4,911 | 0.9631 | 0.9677 | 0.9668 | 0.9191 | 0.9654 |
| 3 | 21,223 | 660 | 248 | 3,901 | 0.9402 | 0.9698 | 0.9651 | 0.8958 | 0.9550 |
| 4 | 22,561 | 589 | 484 | 2,398 | 0.8321 | 0.9746 | 0.9588 | 0.8172 | 0.9033 |
| 5 | 23,227 | 282 | 202 | 2,321 | 0.9199 | 0.9880 | 0.9814 | 0.9056 | 0.9540 |

(Continued)

**Supplementary Table 4:** Continued

| with $Data_{org}$ only | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
|---|---|---|---|---|---|---|---|---|---|
| 6 | 23,265 | 383 | 304 | 2,080 | 0.8725 | 0.9838 | 0.9736 | 0.8583 | 0.9281 |
| 7 | 23,471 | 584 | 321 | 1,656 | 0.8376 | 0.9757 | 0.9652 | 0.7854 | 0.9067 |
| 8 | 23,939 | 74 | 342 | 1,677 | 0.8306 | 0.9969 | 0.9840 | 0.8897 | 0.9138 |
| 9 | 24,198 | 174 | 547 | 1,113 | 0.6705 | 0.9929 | 0.9723 | 0.7553 | 0.8317 |
| 10 | 24,202 | 235 | 366 | 1,229 | 0.7705 | 0.9904 | 0.9769 | 0.8035 | 0.8805 |
| mean | 23,061 | 368 | 368 | 692 | 0.8247 | 0.9839 | 0.9717 | 0.8381 | 0.9043 |
| ISSL-SPR | TN | FP | FN | TP | Sensitivity | Specificity | Accuracy | F1 | BA |
| 1 | 24,233 | 55 | 512 | 1,232 | 0.7064 | 0.9977 | 0.9782 | 0.8129 | 0.8521 |
| 2 | 20,088 | 845 | 188 | 4,911 | 0.9631 | 0.9596 | 0.9603 | 0.9048 | 0.9614 |
| 3 | 21,188 | 695 | 242 | 3,907 | 0.9417 | 0.9682 | 0.9640 | 0.8929 | 0.9550 |
| 4 | 22,620 | 530 | 511 | 2,371 | 0.8227 | 0.9771 | 0.9600 | 0.8200 | 0.8999 |
| 5 | 23,233 | 276 | 269 | 2,254 | 0.8934 | 0.9883 | 0.9791 | 0.8921 | 0.9408 |
| 6 | 23,422 | 226 | 346 | 2,038 | 0.8549 | 0.9904 | 0.9780 | 0.8769 | 0.9227 |
| 7 | 23,573 | 482 | 278 | 1,699 | 0.8594 | 0.9800 | 0.9708 | 0.8172 | 0.9197 |
| 8 | 23,933 | 80 | 370 | 1,649 | 0.8167 | 0.9967 | 0.9827 | 0.8799 | 0.9067 |
| 9 | 24,232 | 140 | 451 | 1,209 | 0.7283 | 0.9943 | 0.9773 | 0.8036 | 0.8613 |
| 10 | 24,220 | 217 | 379 | 1,216 | 0.7624 | 0.9911 | 0.9771 | 0.8032 | 0.8768 |
| mean | 23,074 | 355 | 355 | 692 | 0.8349 | 0.9843 | 0.9728 | 0.8504 | 0.9096 |