Performance Analysis Of Class Rebalancing Self-Training Framework For Imbalanced Semi-Supervised Learning

1st Alvaro Septra Dominggo Nauw
Faculty of Electrical Engineering
Telkom University
Bandung, Indonesia
alvaronauw@student.telkomuniversity.ac.id

2nd Suryo Adhi Wibowo
Faculty of Electrical Engineering
Telkom University
Bandung, Indonesia
suryoadhiwibowo@telkomuniversity.co.id

3rd Casi Setianingsih
Faculty of Electrical Engineering
Telkom University
Bandung, Indonesia
setiacasie@telkomuniversity.ac.id

***Abstract* — This research aims to analyze the effectiveness of the Class-Rebalancing Self-Training (CReST) method in semi-supervised learning (SSL) on class-imbalanced data. The study uses the CIFAR 10 long-tailed dataset to test the performance of SSL with CReST using Python programming language on the Google Colab platform. The results showed that CReST effectively reduces pseudo-labels in the majority class and increases recall in the minority class, with the best performance achieved at Generation 16. However, there was a decrease in Average Accuracy Recall per Class after Generation 16. The study suggests addressing the over-sampling issue and exploring the application of the CReST framework in other areas of machine learning and AI.**

***Kata kunci*— CReST, Semi-Supervised Learning, imbalance data, pseudo label, Semi-Supervised Learning Generation**

I. INTRODUCTION

The field of machine learning has seen significant growth in recent years, becoming the backbone of artificial intelligence and data science [1]. One of the most widely used machine learning algorithms for managing datasets with unlabeled data is semi-supervised learning (SSL) [2]. However, one of the challenges in SSL is dealing with the class imbalance in the dataset, which can cause the model to be biased towards the majority classes and result in lower accuracy for minority classes. Previous research has shown that biased data can result in models that favor majority classes [3]. Class rebalancing techniques have been proposed to overcome the problem of class imbalance in SSL. One such technique is the Class Rebalancing Self-Training (CReST) framework [4], which updates the model by adding the pseudo-labels that it deems most accurate to the correct set of labels while aggressively updating the minority class during self-training iterations. However, the process that occurs in each generation of SSL after the implementation of CReST and the changes in pseudo-labeling in each generation still need to be thoroughly studied.

This research aims to examine the class rebalancing process in each generation of the CReST algorithm and to prove that the minority class is indeed aggressively sampled. We will use the CIFAR-10 Long-tailed dataset, which has an imbalanced class distribution, to determine the best generation of CReST. By applying CReST to SSL with poor performance on imbalanced data, we expect to obtain appropriate generations and good accuracy, precision, and recall. The problem we aim to address is the bias towards majority classes in SSL due to class imbalance in the dataset. Our methodology involves applying the CReST algorithm to SSL and investigating the changes in pseudo-labeling in each generation. We will evaluate the performance of CReST on the CIFAR-10 Long-tailed dataset and compare it with other SSL algorithms. In summary, this research provides insights into the class rebalancing process in each generation of the CReST algorithm and evaluates its performance on imbalanced data. The findings of this study will contribute to improving the accuracy, precision, and recall of SSL algorithms for imbalanced datasets.

II. THEORITICAL REVIEW

A. Research Flow Chart

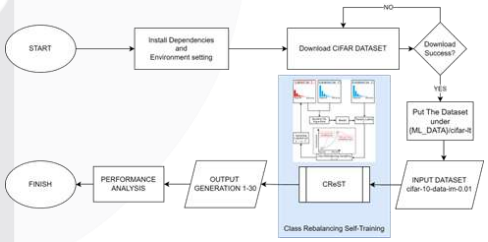

FIGURE 1.
Research flow chart.

The research flow chart in Figure 1. for this research becomes a reference for the writer in analyzing the performance of class rebalancing self-training for imbalance semi-supervised learning. The thing to do is to install dependencies. Environment settings then download and generate the CIFAR datasets, determine the input class imbalance ratio and apply CREST[4] to the CIFAR 10 Long-Tailed datasets, which have an unbalanced class distribution, and then look at the performance of the CReST for every five generations.

B. Long-Tailed Dataset

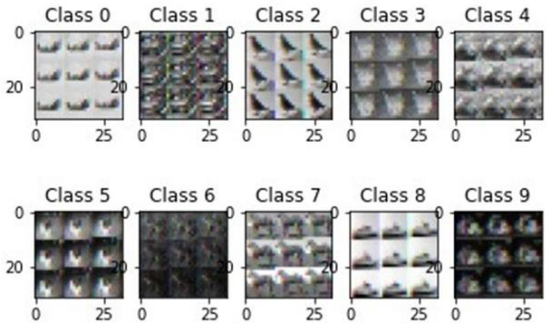

FIGURE 2.
File Train.tfrecords.

A dataset is a collection of data used to train and test a machine learning model. an article written by Bauer and Kohavi stated that the problem that often arises in machine learning modeling is when the dataset used to train the model is not the same as the dataset used to test the model [5]. This is known as a "dataset shift" and can cause a decrease in model performance. The article also highlights the importance of identifying and addressing dataset shifts in machine learning modeling. Therefore, in this research that attempts to analyze the performance of class rebalancing self-training framework for imbalance semi-supervised learning, the dataset that should be used for this research is a dataset that has an unbalanced data distribution, therefore in research chooses a modified cifar dataset so that it has an unequal distribution of data division in each class.

CIFAR (Canadian Institute for Advanced Research) is a dataset used in machine learning research developed by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. CIFAR 10 and CIFAR 100 are variants of the CIFAR dataset used in the study. According to the research "Learning Multiple Layers of Features from Tiny Images" by Alex Krizhevsky [6], CIFAR 10 consists of 60,000 images grouped into ten classes. CIFAR 100 consists of 60,000 images grouped into 100 more specific classes. These two datasets train machine learning models and evaluate model performance.

CIFAR Long-Tailed(LT)is a variant of the CIFAR dataset used to evaluate model performance on unbalanced data. According to the research "Long-tailed Recognition in Large-scale Datasets" by Tong Wu et al. [7], CIFAR Long-Tailed was developed by changing the data distribution from the original dataset to an unequal distribution. It is intended to evaluate model performance on unbalanced data and to evaluate methods that can address the problem of unbalanced data. After downloading the dataset, the contents of the downloaded data are shown in Figure 3 and in Figure 2, which shows one image for each class. The input_shape of the images is $32 \times 32 \times 3$ The class_im_ratio in Figure 3 is used to measure the level of class imbalance in a dataset. This ratio measures the ratio of the number of data in the majority class to the number of data in the minority class. The higher this ratio, the more imbalanced the dataset is. Typically, this ratio is calculated by taking the number of data in the majority class divided by the number of data in the minority class. An example of calculating the class imbalance ratio can be seen in Table 1.

TABLE 1.
Example of the Class im Ratio explanation in Cifar-10-data-im.

Class imbalance ratio	total data in the majority class	calculation	total data in minority class
class im ratio	majority class	majority class x class im ratio	minority class
0.1	5000	5000 ×0.1	500
0.05	5000	5000 ×0.05	250
0.02	5000	5000 ×0.02	100
0.01	5000	5000 ×0.01	50
0.005	5000	5000 ×0.005	25

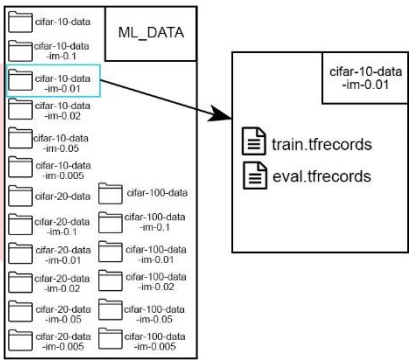

FIGURE 3.
Folder data.

C. Class Rebalancing Self Training using google colab platform

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
%cd /content/drive/MyDrive/Baru/crest
```

```
/content/drive/MyDrive/Baru/crest
```

```
!git clone https://github.com/google-research/crest.git
```

FIGURE 4.
Connect google drive and clone from GitHub.

According to [8], Google Colaboratory, commonly known as Google Colab or simply Colab, is a research project aimed at creating a platform for developing machine learning models with access to high-performance hardware like GPUs and TPUs. It offers a serverless Jupyter notebook environment and is ideal for interactive development. Google Colab is a free resource, just like other G Suite products.

The author's first step in Google Colab, as seen in Figure 4, was to connect it with Google Drive. Before that, the author created a new folder in Google Drive to store the code for the CReST framework for imbalanced SSL, which the author took from the paper "CReST: A Class-Rebalancing Self-Training Framework for Imbalanced Semi-Supervised Learning" that the code already uploaded on GitHub [4]. The author then cloned the GitHub code from the GitHub link and inserted it into the author's drive folder.

1. Install dependencies

The code in Figure 5 is a shell script that installs and sets up a virtual environment for a Python 3 project. The script performs the following tasks:

a. Installs the required packages using the apt package manager: python3 − dev, python3 − virtualenv, python3 − tk, and imagemagick.
b. Installs virtualenv using apt.
c. Creates a virtual environment for Python3 using virtualenv with the − −system −site−packages flag.
d. Activates the virtual environment using the source command.
e. Installs the packages listed in the requirements.txt file using pip.
f. Installs the immutabledict package using pip.

This script ensures that the project has access to all the necessary packages and dependencies while also allowing it to run independently of other projects on the system. The version of TensorFlow used is TensorFlow with version 2.10.0.

```
!apt install python3-dev python3-virtualenv python3-tk imagemagick
!apt install virtualenv
!virtualenv -p python3 --system-site-packages env3
!source env3/bin/activate
!pip install -r requirements.txt
!pip install immutabledict
```

FIGURE 5.
Install Dependencies.

2. Environment Settings

The code in Figure 6 sets environment variables in the terminal. The first line sets the environment variable ML DATA to the directory's path where the machine learning data is stored. The second line sets the environment variable ML DIR to the path of the main directory for the machine learning project. The third line sets the environment variable RESULT to the directory's path where the machine learning project results will be stored. The fourth line adds ML DIR to the PYTHONPAT H, which allows the machine learning code to access modules stored in ML DIR.

```
%env ML_DATA=/content/drive/MyDrive/Baru/crest/ML_DATA
%env ML_DIR=/content/drive/MyDrive/Baru/crest
%env RESULT=/content/drive/MyDrive/Baru/crest/result
%env PYTHONPATH=$PYTHONPATH:$ML_DIR
```

FIGURE 6.
Environment settings.

Running Experiment on Long-tailed CIFAR 10

As shown in Figure 7, the code executes a machine-learning script using the Python programming language. The script is called "train and eval loop" and is executed as a module (-m option) with various arguments passed. The arguments include the following:

a. model dir: the directory where the model will be stored is set to "/content/- drive/MyDrive/Baru/crest/result."
b. method: the method for training the model, which is set to " fixmatch."FixMatch is an algorithm for semi-supervised learning (SSL) leveraging unlabeled data to improve a model's performance. FixMatch combines two standard SSL methods: consistency regularization and pseudo-labeling. The algorithm generates pseudo-labels for unlabeled images using the model's predictions on weakly-augmented images. If the model produces a high-confidence prediction, the pseudo-label is kept. The model is then trained to predict the pseudo-label when fed a strongly-augmented version of the same image [9].
c. dataset: the name of the dataset that will be used for training, which is set to "cifar10lt".
d. input shape: the shape of the input images is set to 32x32x3.
e. class imratio: the ratio of images per class, which is set to 0.01.
f. percent labeled: the percentage of labeled images in the dataset is set to 0.1.
g. num epoch: the number of training epochs is set to 64.
h. num generation: the number of generations for the training process is set to 30.

```
!python -m train_and_eval_loop \
  --model_dir=/content \
  --method=fixmatch \
  --dataset=cifar10lt \
  --input_shape=32,32,3 \
  --class_im_ratio=0.01 \
  --percent_labeled=0.1 \
  --num_epoch=64 \
  --num_generation=30 \
  --do_distalign=false
```

FIGURE 7.
Running experiment.

D. Wide Residual Network

WRN28_w2_bn_leaky_relu_cls10 is utilized in this research for Class Rebalancing Self Training for imbalance semi-supervised learning using the CIFAR-10 long-tailed dataset. This model is a Wide Residual Network (WRN) with 28 weight layers, two widening factors, batch normalization (BN), leaky ReLU activation function, and classifying of ten classes. The utilization of WRN has been proven to achieve high accuracy in image classification tasks [10]. Meanwhile, batch normalization helps improve the training process by reducing the internal covariate shift [11]. Leaky ReLU, as the activation function, helps to overcome the problem of dying ReLU by introducing a slight negative slope [12]. The Class Rebalancing Self Training (CreST) method is applied to tackle the imbalance problem in semi-supervised learning. CreST aims to balance the number of instances in each class while leveraging the unlabeled data [13]. This method has effectively improved performance in imbalanced datasets [14]. The WRN28_w2_bn_leaky_relu_cls10 model with the implementation of the CreST method provides a promising solution for imbalanced semi-supervised learning using the CIFAR-10 long-tailed dataset.

III. METHOD

A. Dataset Changes at Every 5 Generations

Figure 8 presents data from various generations of a dataset to analyze the impact of the Class Rebalancing Self-Training (CReST) framework on addressing the class imbalance in image classification tasks. The first-generation dataset before training revealed that the data distribution needed to be more balanced. There needed to be more than semi-supervised learning (SSL) to generate pseudo-labeled

data in the second generation to fully implement the CReST framework. However, in subsequent generations, the CReST framework was applied, resulting in an increase in data for minority classes and a decrease for majority classes in the pseudo-labeled set. The CReST framework's focus on the minority class, particularly Class 10, was evident in generations 10, 15, 20, 25, and 30, where there was a significant increase in pseudo-labeled data. By generation 30, the majority class gradually decreased the number of pseudo-labeled images. In contrast, the minority class significantly increased, particularly for Class 9, which had the least amount of original data. Overall, the CReST framework effectively addressed the class imbalance in the dataset, resulting in a more balanced data distribution for image classification.

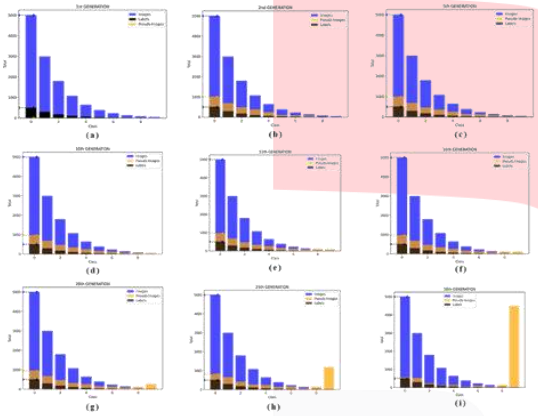

FIGURE 8.
(a) Generation 1, (a) Generation 2, (a) Generation 5, (a) Generation 10, (a) Generation 15, (a) Generation 16, (a) Generation 20, (a) Generation 25, (a) Generation and (a) Generation 30.

B. Precision

Figure 9 is a bar chart that illustrates the precision of each class in each generation, specifically in generations 1, 5, 10, 15, 20, 25, and 30. The chart shows that in generation 1, the majority class (class 1) had lower precision than the minority class (class 9). However, the precision of class 1 gradually improved in each generation, while the minority class experienced a continuous decrease in precision.

FIGURE 9.
Precision in every class (every five generations).

C. Recall

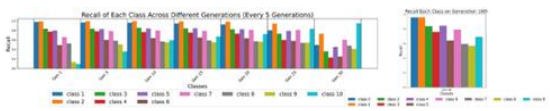

FIGURE 10.
Recall in every class (every five generations).

The results for precision and recall are shown in Figure 10. Interestingly, the recall values are inversely compared to the precision values. The majority class, represented by class 0 in the first generation, has a high recall value of 0.977, while the minority class has a much lower recall value of 0.092. However, the recall value for the minority class consistently increases in each generation, while the majority class experiences a gradual decrease. This further highlights the effectiveness of the CReST framework in addressing the class imbalance issue in the dataset.

D. Average Accuracy Recall per Class

Figure 11 presents the average accuracy recall per class results of using the CReST framework for imbalanced semi-supervised learning. The x-axis displays the generations, and the y-axis represents the accuracy recall per class. The graph shows that accuracy recall per class increases with the number of generations, starting from the 1st generation with an average recall per class of 0.627, reaching its peak at the 16th generation with an average recall per class of 0.768. However, the accuracy recall per class declines at the 17th generation and continues to decrease until the 30th generation with an average recall per class of 0.495.

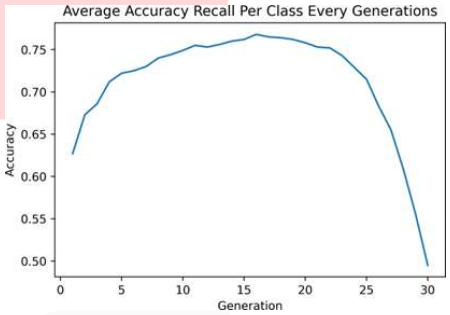

FIGURE 11.
Average Accuracy Recall Per Class for Every Generation.

IV. CONCLUSION

The Class Rebalancing Self Training Framework for imbalanced semi-supervised learning was tested and analyzed on the CIFAR 10LT dataset. The 16th generation was found to be the best, with an Average Accuracy Recall Per Class value of 0.768. The precision and recall values for each class followed the best values, and there was no massive over-sampling. This research found that reducing pseudo-labels in the majority class and increasing them aggressively in the minority class is the key to the framework's success. In the first generation, SSL is applied to generate pseudo-images in each class. With each generation, the data in the majority class decreases while the minority class increases. The precision of the majority class increased in subsequent generations, while the minority class's recall improved after the application of CReST. However, the precision in the minority class decreased with each generation. Further research is suggested to address the oversampling issue by using the Distribution Alignment method or other methods that can handle over-sampling of the minority class during increasing generations. The CReST framework can also be explored in other areas of machine learning and artificial intelligence to increase its impact. Another suggestion is to add an algorithm that selects the best generation in applying CReST on imbalanced SSL. Experiments using the Mixmatch method can be conducted, and further optimization of the CReST framework, including selecting appropriate parameters and hyperparameters, is required to improve its effectiveness.

REFERENCE

[1] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis lectures on artificial intelligence and machine learning*, vol. 3, no. 1, pp. 1–130, 2009.

[2] T. O. Ayodele, "Types of machine learning algorithms," *New advances in machine learning*, vol. 3, pp. 19–48, 2010.

[3] M. C. du Plessis and M. Sugiyama, "Semi-supervised learning of class balance under class-prior change by distribution matching," *Neural Networks*, vol. 50, pp. 110–119, 2014.

[4] C. Wei, K. Sohn, C. Mellina, A. L. Yuille, and F. Yang, "CReST: A Class-Rebalancing Self-Training Framework for Imbalanced Semi-Supervised Learning," *CoRR*, vol. abs/2102.09559, 2021, [Online]. Available: https://arxiv.org/abs/2102.09559

[5] E. Bauer and R. Kohavi, "A survey of dataset shift in machine learning," *The Journal of Machine Learning Research*, vol. 8, pp. 1825–1832, 2007.

[6] A. Krizhevsky, G. Hinton, and others, "Learning multiple layers of features from tiny images," 2009.

[7] T. Wu, X. Wang, Q. Ye, and Y. Liu, "Long-tailed recognition in large-scale datasets," *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 816–833, 2018.

[8] E. Bisong and E. Bisong, "Google colaboratory," *Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners*, pp. 59–64, 2019.

[9] K. Sohn, R. Roelofs, B. Zoph, Q. v Le, and J. Shlens, "FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence," *arXiv preprint arXiv:2001.07685*, 2020.

[10] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 778–786.

[11] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[12] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," *Proc. ICML*, vol. 30, p. 3, 2013.

[13] Z. Zhou, L. Chen, J. Sun, and X. Wang, "Deep Class Rebalancing for Long-Tailed Recognition," *IEEE Trans Pattern Anal Mach Intell*, 2020.

[14] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao, "Range Loss for Deep Face Recognition with Long-tail," Nov. 2016, [Online]. Available: http://arxiv.org/abs/1611.08976

..