COMP 433 Project Report

Jeremy Ouellette (ID: 40212872), Mazen Mahmoud (ID: 40175486)

Fall 2023

1 Introduction

Label noise refers to inaccuracies or errors present in the assigned labels of a dataset used to train supervised machine learning models. In an ideal scenario, each data point (sample) should bear a correct label, guiding the model during training. However, real-world datasets often exhibit label noise, where certain data points carry incorrect or noisy labels. This poses a significant challenge for supervised learning models, as they depend on accurate labels to learn and to generalize patterns effectively. Label noise can originate from various sources, including human annotation errors and ambiguous instances that prove challenging to label. This issue is particularly pronounced in datasets designed to challenge humans, such as those from CAPTCHA systems, making them prone to label noise.

However, despite the common practice of using deeper Convolutional Neural Networks (CNNs) to address complex tasks as is the case here, this approach falls short in handling label noise complexities. Indeed, only increasing model depth may worsen the problem by causing overfitting to noisy labels, reducing the model's ability to discern real patterns.

This study aims to research the advantages of two specific techniques designed to increase resilience against label noise by comparing their outcomes with a baseline ResNet-18 CNN model. The report includes a comprehensive description of the noisy dataset in use, a thorough exploration of previous solutions through a literature study, an examination of the baseline model's characteristics and results, an in-depth analysis of each proposed technique's impact, and an overall comparison of the results.

2 Dataset

The CIFAR-10 dataset comprises 60,000 32x32 color images distributed across 10 classes, with 6,000 images per class. It is divided into 50,000 training images and 10,000 test images. It includes the following classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. These classes are mutually exclusive, with no overlap between automobiles and trucks. The "automobile" category encompasses sedans, SUVs, and similar vehicles, while the "truck" category includes only large trucks, excluding pickup trucks [1].

2.1 Introduction of Label Noise

For this study, the CIFAR-10 dataset has been adapted to incorporate potentially noisy labels. The algorithms are designed to train on this modified dataset, taking into account two types of label noise:

- **Symmetric Label Noise**

  Symmetric label noise introduces uncertainty into the training data by flipping correct labels with a probability ϵ. For instance, if there are 100 bird images in the training set and the noise level ϵ, then $100(1-\epsilon)$ of these images will retain the label "bird," while 100ϵ will be randomly assigned a label different from "bird."

- **Asymmetric Label Noise**

  Asymmetric label noise introduces label-flipping following predefined rules. As per the project requirements, the explicit rules

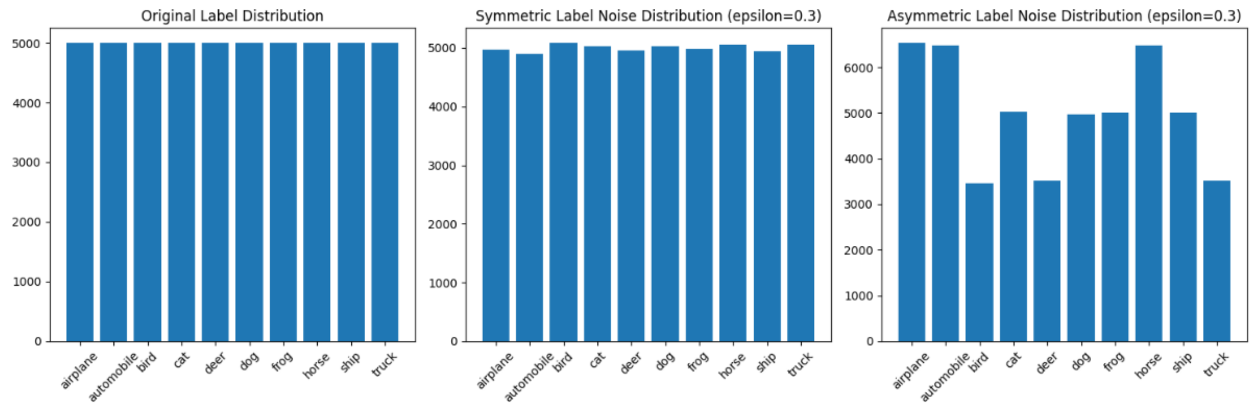


Figure 1: Training set class distribution. The distribution is less uniform in the second and third graphs due to label noise.

for label flipping are as follows: ”truck” should be flipped to ”automobile,” ”bird” to ”airplane,” ”deer” to ”horse,” ”cat” to ”dog,” and ”dog” to ”cat.” The determination of this flipping probability is tied to the asymmetric noise level ϵ. To illustrate, if there are 100 bird images in the training data, $100(1-\epsilon)$ of these should maintain the original ”bird” label, while the remaining 100ϵ bird images will undergo a label flip, changing their label to ”airplane.”

Both symmetric and asymmetric label noise present challenges during training. On one side, symmetric noise introduces a stochastic noise as it mixes randomly the labels for all classes with all other classes. On the other side, asymmetric noise skews the learning in only one other way but more significantly. Figure 1 summarizes the distribution of the training set from the original dataset and from its two derivatives.

3 Literature Study

To overcome the reduction in accuracy due to label noise, a large scope of strategies are available. These encompass the development of more robust architectures [4] [20], the usage of specialized loss functions [7] [11] [22], selective sampling techniques [15] [6], and others [15]. The next subsection will do a brief overview of the three main strategies that were researched in this work.

3.1 Main Strategies

3.1.1 Noise Resilient Architectures

To avoid the burden of dealing with noisy labels in the first place, it is possible to use unsupervised machine learning techniques that do not require any labels during training. Popular unsupervised models are k-means clustering [4] and probability density estimation [20], where the former does not allow samples to be members of more than one category, but the latter does. These strategies use the insight that samples like images that have similar features are likely to be in the same category, and all that is needed is to know beforehand the number of categories. However, these techniques are not fit to be trained directly on CIFAR-10 as each image, should be embedded first into a feature vector created with the help of a trained CNN model (or any other model capable of this task) where its last classification layer is removed [12].

Semi-supervised machine learning techniques are also an interesting alternative implemented in various works [18]. They allow the model to be trained on a small sample of correctly labeled data and then on the rest of the data by making an abstraction of their noisy labels. This requirement could, however, not be met in this report as the noise is applied to the entire CIFAR-10 dataset.

Another promising approach exploited recently is to leverage early learning. Indeed, in the first epochs, DL models tend to learn features that generalize better compared to the learning done in the last epochs which tend to overfit more to the noise. Recent SOTA architectures leverage this knowledge to increase their robustness against noise [10] [5].

3.1.2 Robust Loss Function:

Choosing the right loss functions is crucial in making machine learning (ML) models strong against noise. In the last years, there has been a rise in interest in normalized loss functions as it was proven to increase robustness if a majority of labels in a class are correctly labeled [7] [11]. However, this effort to boost robustness sometimes leads to a new problem called underfitting. Even though well-known loss functions like cross-entropy and focal loss become resilient with normalization, it doesn't always make the models more accurate overall. In fact, normalized cross-entropy and normalized focal loss usually perform even worse than their regular versions [11]. Because of this interest in hybrid loss functions merging both traditional, fast-converging loss and normalized more resilient loss rose [19] [11]. One popular example is ‘NCEandRCE‘, combining Normalized Cross-Entropy (NCE) and Reverse Cross-Entropy (RCE) where the final output is a linear combination of both [11]. Another interesting loss is called ’Bootstrapping’ where prior knowledge about the noise ratio is used in the loss function to overfit less to noise [2].

3.1.3 Sample Selection

Sample selection is concerned with locating noisy samples during training and reducing their impact. Common techniques follow:

- **Outlier Detection [14] [6]:**
  - Identify samples deviating significantly in feature space.

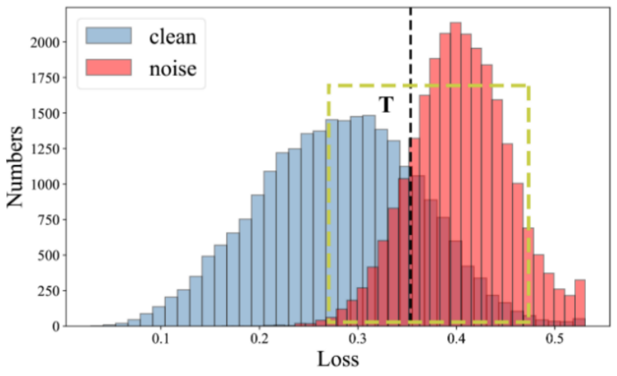


Figure 2: The distribution of losses for both clean and noisy samples in CIFAR-100, where 50% of the noise is symmetrically introduced. Extracted from PGDF paper [6].

  - Utilize statistical measures or machine learning models for outlier detection.
  - Exclude or down-weight identified outliers during training.
  - See Figure 2
- **Consistency Checks:**
  - Train multiple models on different data subsets.
  - Evaluate the consistency of predictions across models for each sample.
  - Consider samples with inconsistent predictions as potentially noisy.
- **Confidence Estimation: [20]**
  - Utilize models providing confidence estimates for predictions.
  - Set a threshold on prediction confidence to identify potentially noisy samples.
- **Transfer Learning [9]:**
  - Leverage pre-trained models on clean data to identify noisy samples.
  - Fine-tune the model on the new dataset, excluding or down-weighting noisy samples.

4 ResNet18 Baseline

ResNet, introduced in 2015, stands out as a simple yet powerful baseline for its scalable architecture and innovative skip connections [8]. This addresses challenges in training very deep models, a limitation faced by earlier architectures like VGG [13] and GoogLeNet [16]. Its impact is evident in subsequent models such as ResNeXt [21] and EfficientNet [17], all adopting the concept of skip connections for improved scalability and performance. Even in its lightweight form, ResNet18 maintains simplicity while striking a balance between model complexity and efficiency, making ResNet a foundational choice in modern computer vision tasks. The ResNet block is illustrated in Figure 8 in the appendix.

Hyperparameter Search

We conducted a hyperparameter search for optimal model performance, experimenting with batch sizes of 64, 128, 256, and 512. Learning rates were adjusted accordingly, starting at 0.001 for a batch size of 64 and scaling for larger batches (e.g., 0.002 for 128, 0.004 for 256). Surprisingly, a learning rate of 0.006 performed better for a batch size of 512, compensating for fewer weight updates in larger batches. This scaling prevents an increase in epochs needed for convergence. Optimal results were observed at 10 epochs, as ResNet18 with standard Categorical Cross Entropy (CE) loss showed signs of overfitting beyond this point. We chose the ADAM optimizer due to faster convergence than SGD. Despite testing ResNet34, its longer training and inference times, driven by a larger number of weights, made it impractical within our time constraint for model training.

For a balanced approach considering computational efficiency and model accuracy, we selected a batch size of 256 and a learning rate of 0.004. This choice strikes a compromise between computational cost, training speed, and achieving satisfactory accuracy. Results in the appendix showcase an alternative configuration with a learning rate of 0.001 and a batch size of 64 4. While it exhibits higher training accuracy, the improvement in test accuracy is limited. Hence, the chosen configuration with a batch size of 256 and a learning rate of 0.004 stands as a practical compromise, ensuring both accuracy and fast training time.

Result Analysis

The outcomes, detailed in Table 1 and summarized in Figure 3, showcase a notable decline in test accuracy as label noise intensifies, dropping from 78.13% to 10.44% in symmetric conditions (0.1 to 0.9). A similar pattern is evident in asymmetric noise conditions. Intriguingly, results for asymmetric noise exhibit more promise, with test accuracy at the highest noise level surpassing expectations. These findings highlight the inherent difficulty for our baseline model to sustain accuracy in the face of increased label noise, underscoring the necessity to explore more advanced solutions. On a positive note, the model demonstrates swift inference times, fluctuating between 0.0035 and 0.005 seconds, while training time for each noise level records around 200 seconds, utilizing a 32-bit precision V100 GPU on Google Colab. An alternative hyperparameter configuration is shown in the appendix 4.

ϵ	**Symmetric**		**Asymmetric**	
	Train	**Test**	**Train**	**Test**
0.1	63.77	76.92	70.95	77.32
0.3	44.38	71.31	64.45	73.50
0.5	29.40	66.80	59.64	59.57
0.8	10.96	21.89	67.65	46.08
0.9	10.00	9.83	71.54	46.73

Table 1: Baseline Train and Test Accuracies for Symmetric and Asymmetric Noise (batch_size = 256 and learning_rate = 0.004) and the output in percentage.

5 Proposed Solution Overview

5.1 ResNet18 with SCE Loss

The first strategy we implemented to counter accuracy reduction related to label noise is to replace the categorical Cross-Entropy (CE) func-

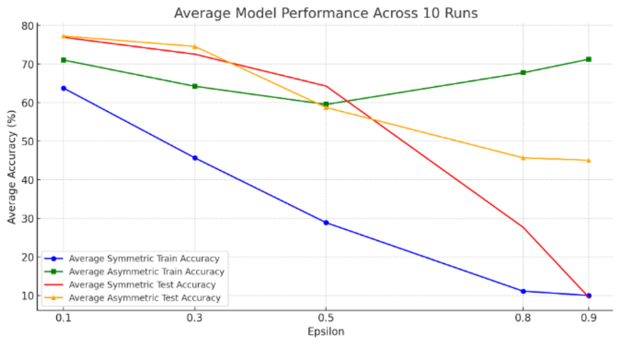
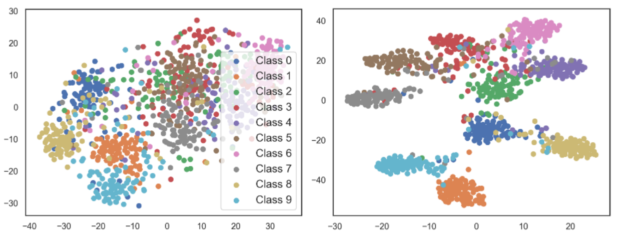


Figure 3: Baseline ResNet18 Results.



(a) CE (b) SL

Figure 4: Representations acquired by CE and SL on the CIFAR-10 dataset containing 60% symmetric noisy labels. Figure extracted from the SL paper [19]

tion with the Symmetric Cross-Entropy Loss (SCE Loss).

SCELoss: Symmetric Cross-Entropy Loss

The `SCE Loss` is a custom loss function designed to handle label noise in classification tasks. Unlike some methods that use normalized loss functions and risk underfitting, SCE avoids underfitting by not using normalization at all. Instead, it supplements the traditional Cross Entropy Loss (CE) with Reverse Cross Entropy Loss (RCE). This balanced approach aims to enhance robustness without sacrificing accurate model predictions in the presence of label noise. This strategy was originally introduced in this work [19] under the name SL. We were inspired to implement this solution due to its ability to extract less noisy features from CIFAR-10 as showcased in Figure 4 where the clusters in (b) are way purer compared to (a).

Cross-Entropy Loss (CE)

Traditional Cross-Entropy Loss is a widely used metric in classification tasks. It quantifies the disparity between the predicted probability distribution and the true distribution of the labels. Mathematically, CE is defined as the negative log-likelihood of the true class:

$$\mathrm{CE}(p, q) = -\sum_i p_i \cdot \log(q_i)$$

Here, p represents the true probability distribution (one-hot encoded labels), and q denotes the predicted probability distribution generated by the model.

Reverse Cross-Entropy Loss (RCE)

Reverse Cross-Entropy Loss is introduced as a modification to traditional CE loss to enhance the model's robustness to label noise. RCE reverses the role of the true and predicted distributions, emphasizing correct predictions for instances where the true class probability is high:

$$\mathrm{RCE}(p, q) = -\sum_i q_i \cdot \log(p_i)$$

Here, p represents the predicted probability distribution, and q is the true probability distribution (one-hot encoded labels).

Merging CE and RCE

The `SCELoss` combines traditional CE and RCE, introducing a weighted sum of both losses. The parameters α and β control the contribution of each term. The final SCE loss is given by:

$$\mathrm{SCELoss} = \alpha \cdot \mathrm{CE} + \beta \cdot \mathrm{RCE}$$

The motivation behind SCE is to leverage the strengths of both CE and RCE to effectively handle label noise. CE focuses on correctly classifying instances where the predicted probability is high, while RCE emphasizes correct predictions when the true class probability is high.

The additional `clamp` operations applied to the predicted probabilities and one-hot encoded labels in the code are used to prevent numerical instability, ensuring numerical robustness during computation.

Hyperparameter Search

We used the same batch size and learning rate combinations as in the baseline, but as the SCE loss converged a bit slower than CE with ADAM, we increased the number of weight updates by setting the batch size to 64 and the learning rate to 0.001. For α and β, both were set to 1 as recommended in the SCE paper, meaning equal weights for CE and RCE in the final output. As there were no signs of SCE overfitting to noise after 10 epochs, we extended training to 15 epochs. While a higher number could have been beneficial, resource constraints limited the training time.

Result Analysis

ResNet18+SCE results are showcased in Table 2 and Figure 5. For all symmetric noise levels, the improvements in test accuracy in percentage points are as follows: 1.63, 4.66, 5.65, 11.71, and -0.59. It is clear that as noise increases, improvements over the baseline become more pronounced. For asymmetric noise levels, the changes are relatively small: 1.3, 4.7, -0.79, -0.55, -1.31. The inference time is approximately the same as the baseline and the training time for each noise level was around 370 seconds on Colab's V100 with 32-bit precision.

Additional Motivation

Our decision to implement the SCE loss comes from extensive testing of other alternatives. We implemented and trained with L_DMI, NCE+RCE, Normalized CE, Normalized RCE, Generalized CE, Mean Absolute Error (MAE), label smoothing-based loss, and bootstrapping-based loss. In all of these losses, with no exception, we experienced underfitting with accuracy often under 30% on the training set for a symmetric noise level of 0.1 and 10 epochs. We tested each of them with ResNet18, ResNet34, EfficientNet-B0, EfficientNet-B1, and EfficientNet-B3 with no improvement for any of them. We believe that as most of these methods converge much slower compared to CE, they would have benefited from way longer training (100 epochs). Indeed, even with SCE not using any normalization hindering convergence speed, 15 epochs were still a serious limitation, as can be seen in Figure 6. For this reason, the SCE loss is, to our knowledge, the best balance between training time, inference time, and test accuracy.

ϵ	**Symmetric**		**Asymmetric**	
	Train	**Test**	**Train**	**Test**
0.1	77.57	78.55	83.09	78.62
0.3	57.84	75.97	74.42	78.20
0.5	39.08	72.45	67.94	58.78
0.8	12.69	33.60	78.88	45.53
0.9	9.97	9.24	83.40	45.42

Table 2: ResNet18+SCE Train and Test Accuracies for Symmetric and Asymmetric Noise (batch_size = 64 and learning_rate = 0.001) and the output in percentage.

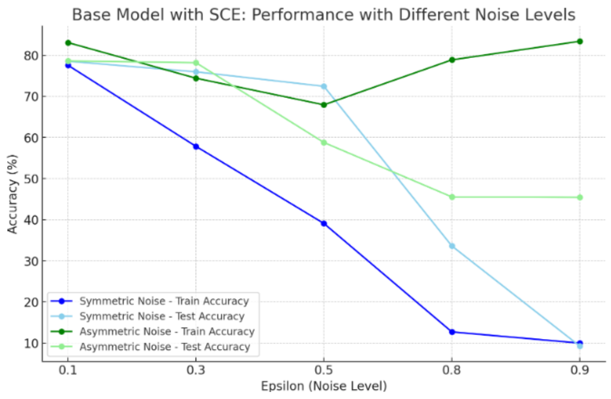


Figure 5: RestNet18 with SCE Loss Results

5.2 EfficientNet-B3 with SCE Loss and Transfer Learning

The second strategy to increase resilience against label noise implemented for this work was to replace resnet18 with a pre-trained EfficientNet-B3 with the Symmetric Cross-Entropy Loss

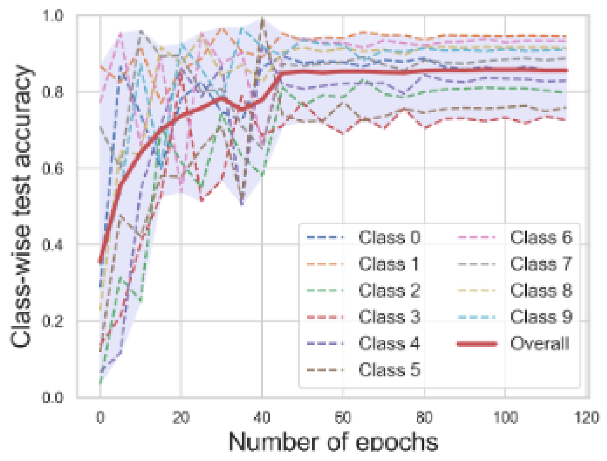


Figure 6: ResNet34 accuracy wrt. epochs on CIFAR-10 (40% symmetric label noise). Extracted from the SCE paper [19].

(SCELoss).

EfficientNet-B3 Characteristics

EfficientNet-B3 was developed using Neural Architecture Search (NAS), an automated method for designing neural network structures. NAS explores a vast space of possible architectures to find an optimal configuration for a specific task. In contrast, ResNet18 is a manually designed architecture with predefined layer configurations. While the ResNet family is a well-established architecture, the EfficientNet family often outperforms it in terms of performance and model size efficiency as showcased in this figure 9. On top of its better parameter efficiency, EfficientNet-B3 was chosen as it featured a similar number of parameters compared to ResNet18 (slightly lower) as it seemed like a fairer comparison with the baseline, and as it is available pre-trained on the ImageNet dataset.

Motivation

The main motivation behind this decision comes from the influence of recent works like ELR [10] demonstrating that only the early part of the training truly helps the models to generalize well to unseen data and after a moment the learning will mostly train the model to overfit to the noise. Therefore, based on this insight, it makes a lot of sense to use a pre-trained model and fine-tune it for as little as possible on the noisy training data while still using the SCE loss that provides some protection against the noise. It is worthwhile to note that other strategies to limit the impact of noise were tested but were deemed unpractical given that inference time and training time were required to be low as will be discussed in the discussion section below.

Hyperparameter Search

Due to its design, despite having fewer learnable weights than the baseline, EfficientNet-B3 took a considerable amount of time to train. For this reason, we tested only with batch sizes of 256 and 512, 15 epochs, and with learning rates of 0.002, 0.004, and 0.008. The best combination was found to be (256, 0.002). Given its pretrained nature, the model underwent training only once, as its initial weights are not randomly defined.

Result Analysis

EfficientNet-B3+SCE's results are showcased in Table 3 and Figure 7. For all symmetric noise levels, the improvements in test accuracy in percentage points compared to the baseline are as follows: 8.12, 8.41, 14.34, -7.51, and 0.37. It is clear that at noise level ≤ 0.5, EfficientNet-B3 offers a significant improvement over the baseline. For asymmetric noise levels, the changes are major and strictly positive: 8.18, 10.24, 2.16, 1.99, and 0.82. The inference time is approximately 0.005 seconds and the training time for each noise level is around 300 seconds on Colab's V100 with 32-bit precision.

6 Discussion

In this section, we discuss an unsuccessful strategy that we implemented but that provides valuable insights into the usage of unsupervised learning for this task. We'll also cover the current state-of-the-art approach and summarize

ϵ	Symmetric		Asymmetric	
	Train	**Test**	**Train**	**Test**
0.1	76.20	85.04	80.90	85.50
0.3	52.74	79.72	73.19	83.74
0.5	39.35	81.14	66.10	61.73
0.8	10.93	14.38	77.01	48.07
0.9	9.98	10.20	81.41	47.55

Table 3: Train and Test Accuracies for Symmetric and Asymmetric Noise (batch_size = 256 and learning_rate = 0.002) and the output in percentage.

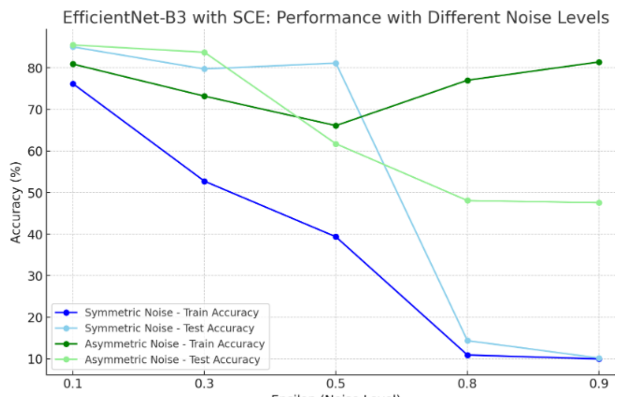


Figure 7: Fine-Tuned EfficientNet-B3 with SCE Loss Results

the strengths of our two proposed strategies designed to handle label noise.

Failed Attempt: CNN-KNN Hybrid Strategy Against Label Noise

Our initial exploration involved a hybrid CNN-KNN (Convolutional Neural Network - k-Nearest Neighbors) model, merging CNN's feature extraction with KNN's unsupervised classification capabilities [12]. The CNN component learned hierarchical representations through convolutional and pooling layers, and the extracted feature embeddings served as data points in a high-dimensional space. However, the model faced limitations on the noisy CIFAR-10. The success of the CNN-KNN approach relies on the separability of features, and when features between classes are too similar or overlapping, the model's performance may be hindered. This is the most probable reason behind the low performance.

Current SOTA on CIFAR-10N

Examining the current state-of-the-art in handling label noise on CIFAR-10 datasets, the Prior Guided Denoising Framework (PGDF) stands out [3]. PGDF employs a two-step strategy, initially segregating samples based on training loss and then using semi-supervised methods for pseudo-label generation. Despite impressive performance gains, PGDF's extended training times and inference time did not meet the constraints of this project.

Applicability of the Proposed Strategies

The SCE Loss presents a versatile solution applicable to various CNN models without significant modifications, demonstrating broad usability. Similarly, the transfer learning technique offers a straightforward way to enhance adaptability by leveraging pre-trained models. These strategies can easily complement other techniques, such as selective sampling, contributing to a robust framework for addressing label noise.

7 Limitations

Our strategies, ResNet18+SCE Loss and transfer learning with EfficientNet-B3+SCE Loss demonstrate limited accuracy improvement with noise levels above 50%. Also to achieve a more optimal performance on noise levels under 50% longer training time would be beneficial. However, practical constraints on computational resources and time limit the feasibility of extending training time.

8 Conclusion

In conclusion, our investigation focused on two strategies: ResNet18 with Symmetric Cross Entropy Loss (SCE Loss) and transfer learning using EfficientNet-B3 with SCE. Both strategies demonstrated improvements in test accuracy for both symmetric and asymmetric noise, achieving up to 11.71% above the baseline for ResNet18+SCE and 14.34% for EfficientNet-B3+SCE. Transfer learning with EfficientNet-B3 showed promise by leveraging pre-trained models to mitigate overfitting to noisy labels. However, practical constraints, including limited training time and computational resources, constrained the depth of our exploration. Despite these limitations, these strategies provide accessible alternatives and valuable insights for building robust models in the presence of label noise.

References

[1] CIFAR-10 and CIFAR-100 datasets. URL: https://www.cs.toronto.edu/~kriz/cifar.html. 1

[2] GJS/losses.py at main · ErikEnglesson/GJS. URL: https://github.com/ErikEnglesson/GJS/blob/main/losses.py. 3

[3] Papers with Code - CIFAR-100N Benchmark (Learning with noisy labels). URL: https://paperswithcode.com/sota/learning-with-noisy-labels-on-cifar-100n. 8

[4] Dara Bahri, Heinrich Jiang, and Maya Gupta. Deep k-NN for Noisy Labels, April 2020. arXiv:2004.12289 [cs, stat]. URL: http://arxiv.org/abs/2004.12289. 2

[5] Yingbin Bai, Erkun Yang, Bo Han, Yanhua Yang, Jiatong Li, Yinian Mao, Gang Niu, and Tongliang Liu. Understanding and Improving Early Stopping for Learning with Noisy Labels, December 2021. arXiv:2106.15853 [cs]. URL: http://arxiv.org/abs/2106.15853. 3

[6] Wenkai Chen, Chuang Zhu, Yi Chen, Mengting Li, and Tiejun Huang. Sample Prior Guided Robust Model Learning to Suppress Noisy Labels, January 2022. arXiv:2112.01197 [cs]. URL: http://arxiv.org/abs/2112.01197. 2, 3

[7] Erik Englesson and Hossein Azizpour. Generalized Jensen-Shannon Divergence Loss for Learning with Noisy Labels, October 2021. arXiv:2105.04522 [cs, stat]. URL: http://arxiv.org/abs/2105.04522, doi:10.48550/arXiv.2105.04522. 2, 3

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, December 2015. arXiv:1512.03385 [cs]. URL: http://arxiv.org/abs/1512.03385. 4, 10

[9] Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. CleanNet: Transfer Learning for Scalable Image Classifier Training with Label Noise, March 2018. arXiv:1711.07131 [cs]. URL: http://arxiv.org/abs/1711.07131, doi:10.48550/arXiv.1711.07131. 3

[10] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-Learning Regularization Prevents Memorization of Noisy Labels, October 2020. arXiv:2007.00151 [cs, stat]. URL: http://arxiv.org/abs/2007.00151, doi:10.48550/arXiv.2007.00151. 3, 7

[11] Xingjun Ma, Hanxun Huang, Yisen Wang, Simone Romano, Sarah Erfani, and James Bailey. Normalized Loss Functions for Deep Learning with Noisy Labels, June 2020. arXiv:2006.13554 [cs, stat]. URL: http://arxiv.org/abs/2006.13554, doi:10.48550/arXiv.2006.13554. 2, 3

[12] Zarin Anjuman Sejuti and Md Saiful Islam. A hybrid CNN–KNN approach for identification of COVID-19 with 5-fold cross validation. *Sensors International*, 4:100229, January 2023. URL: https://www.sciencedirect.com/science/article/pii/S2666351123000037, doi:10.1016/j.sintl.2023.100229. 2, 8

[13] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition (VGG), April 2015. arXiv:1409.1556 [cs]. URL: http://arxiv.org/abs/1409.1556, doi:10.48550/arXiv.1409.1556. 4

[14] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Robust Learning by Self-Transition for Handling Noisy Labels. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1490–1500, August 2021. arXiv:2012.04337 [cs]. URL: http://arxiv.org/abs/2012.04337, doi:10.1145/3447548.3467222. 3

[15] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from Noisy Labels with Deep Neural Networks: A Survey, March 2022. arXiv:2007.08199 [cs, stat]. URL: http://arxiv.org/abs/2007.08199. 2

[16] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), February 2017. Number: 1. URL: `https://ojs.aaai.org/index.php/AAAI/article/view/11231`, `doi:10.1609/aaai.v31i1.11231`. 4

[17] Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, September 2020. arXiv:1905.11946 [cs, stat]. URL: `http://arxiv.org/abs/1905.11946`. 4, 10

[18] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2691–2699, Boston, MA, USA, June 2015. IEEE. URL: `http://ieeexplore.ieee.org/document/7298885/`, `doi:10.1109/CVPR.2015.7298885`. 2

[19] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric Cross Entropy for Robust Learning with Noisy Labels, August 2019. arXiv:1908.06112 [cs, stat]. URL: `http://arxiv.org/abs/1908.06112`. 3, 5, 7

[20] Shuyin Xia, Longhai Huang, Guoyin Wang, Xinbo Gao, Yabin Shao, and Zizhong Chen. An adaptive and general model for label noise detection using relative probabilistic density. *Knowledge-Based Systems*, 239:107907, March 2022. URL: `https://www.sciencedirect.com/science/article/pii/S0950705121010637`, `doi:10.1016/j.knosys.2021.107907`. 2, 3

[21] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated Residual Transformations for Deep Neural Networks, April 2017. arXiv:1611.05431 [cs]. URL: `http://arxiv.org/abs/1611.05431`. 4

[22] Yilun Xu, Peng Cao, Yuqing Kong, and Yizhou Wang. L_dmi: An Information-theoretic Noise-robust Loss Function, November 2019. arXiv:1909.03388 [cs, stat]. URL: `http://arxiv.org/abs/1909.03388`, `doi:10.48550/arXiv.1909.03388`. 2

A Appendix

A.1 Additional Testing

ϵ	**Symmetric**		**Asymmetric**	
	Train	**Test**	**Train**	**Test**
0.1	75.33	77.93	81.36	78.13
0.3	55.28	74.13	73.10	75.81
0.5	36.05	67.59	66.58	58.56
0.8	11.16	21.61	77.14	46.75
0.9	10.10	10.44	81.48	45.82

Table 4: Baseline Train and Test Accuracies for Symmetric and Asymmetric Noise (batch_size = 64 and learning_rate = 0.001).

A.2 Additional Figures

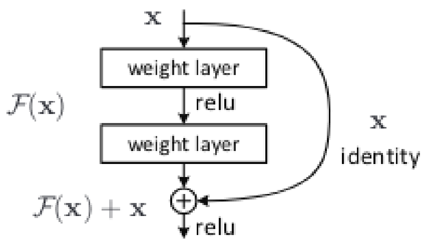
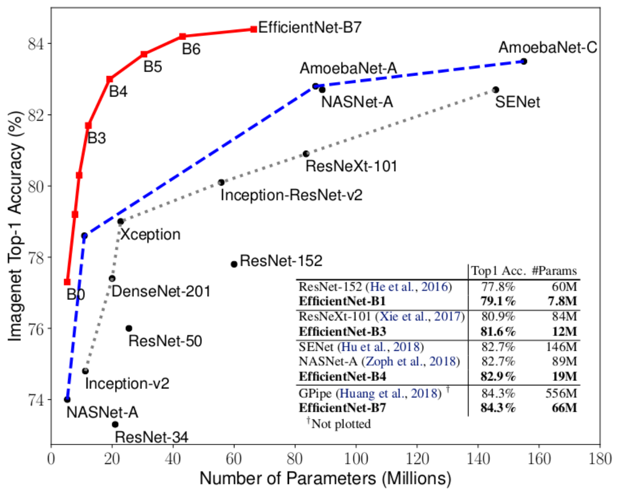

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 8: ResNet Block [8]



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

†Not plotted

Figure 9: Figure extracted from the EfficientNet paper [17].