# Computer Vision with Real-World Data

| | |
|---|---|
| Student | Damir Nurtdinov |
| Student's time capacity | $18 \pm 3$ hours |
| Simulation time capacity (GPU: 20 GB VRAM) | $42 \pm 5$ hours |

## 1 Abstract

This work evaluates three loss functions—cross-entropy (CE) loss, B-loss, and N-loss—across CNN, ResNet50, and ViT architectures on the CIFAR-10N dataset. I compare two ensembling techniques: majority voting (EMV) and confidence-based weighted predictions (EWP). A total of 63 models were trained, and 26 experiments were conducted to assess individual and ensemble performance. Results show that N-loss excels on clean labels (85.89% accuracy for CNN), while B-loss performs best on noisy labels (72.64%). Ensembling improved accuracy, with EWP slightly outperforming EMV. The full ensemble achieved 79.98% accuracy, demonstrating the benefits of combining diverse models and loss functions.

## 2 Introduction

In machine learning, the challenge of noisy labels is a significant obstacle to achieving robust and generalizable models. Noisy labels, which can arise from human error, ambiguous data, or automated labeling processes, often degrade model performance and hinder real-world applications. To address this issue, researchers have explored various approaches, including the development of noise-robust loss functions and ensemble methods. In this work, we investigate the effectiveness of three loss functions—cross-entropy (CE) loss, B-loss, and N-loss—across three distinct architectures—CNN, ResNet50, and ViT—and evaluate their performance using two ensembling techniques: majority voting and confidence-based weighted predictions.

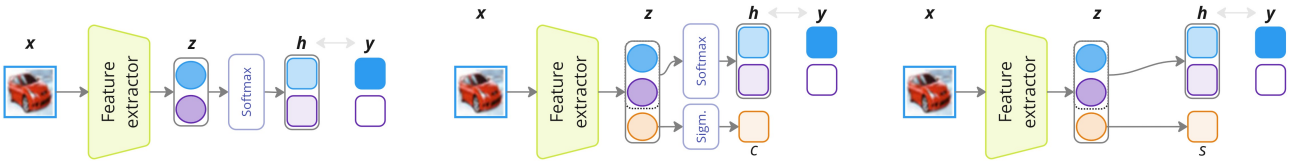
Cross-entropy (CE) loss is the most commonly used loss function for classification tasks, measuring the discrepancy between predicted probabilities and true labels. While effective for clean datasets, CE loss is highly sensitive to label noise, often leading to overfitting on incorrect labels. To mitigate this, I explore two noise-robust loss functions: B-loss and N-loss. B-loss is designed to reduce the impact of noisy labels by downweighting uncertain or potentially mislabeled samples, while N-loss explicitly models the noise distribution and adjusts the learning process to account for label uncertainty. These loss functions provide alternative strategies for handling noisy data, each with its own strengths and trade-offs.

In addition to evaluating individual models, I explore the benefits of ensembling multiple models to improve robustness and accuracy. The first ensembling technique, majority voting (EMV), aggregates predictions from multiple models by selecting the class predicted most frequently. This approach leverages the diversity of individual models to reduce the impact of errors made by any single model. The second technique, confidence-based weighted predictions (EWP), incorporates the confidence scores of each model's predictions. Here, the final prediction is determined by the class with the highest aggregated confidence, giving more weight to predictions from models that are more certain. This method is particularly effective for noisy datasets, as it prioritizes reliable predictions over uncertain ones.

The primary goal of this work is to systematically evaluate the performance of CE loss, B-loss, and N-loss across CNN, ResNet50, and ViT architectures, both individually and in combination with majority voting and confidence-based weighted predictions. By training 63 distinct models and conducting 26 experiments, I aim to answer several key questions: How do majority voting and confidence-based weighted predictions compare in improving model robustness? And what is the impact of combining multiple architectures and loss functions in a full ensemble? Through this study, I aim to provide insights into noise-robust machine learning and offer practical guidance for researchers and practitioners working with noisy datasets.

## 3 Related work

**Learning with Real-World Human Annotations.** *Please analyze in more detail 2–4 papers related to the CIFAR-10N dataset [24], focusing on models and methods that can be implemented in this work for comparison or to enhance the results. Emphasis should be placed on approaches that have the potential to improve the outcomes*




(a) CE-model (b) B-model (c) N-model

Figure 1: Schematics of the classification models under study: the CE-model serves as the baseline (see Equation (11)), the B-model includes an additional output $c \in (0, 1)$ that calculates the certainty value of the prediction $\mathbf{h}$ (see Equation (7)), and the N-model introduces an additional output $s \in (-\infty, \infty)$ that computes the logarithmic variance of the prediction (see).

*of this study. Combine a descriptive approach with a critical analysis. Additionally, ensure that all methods, ideas, and concepts that are not your own are properly referenced.*

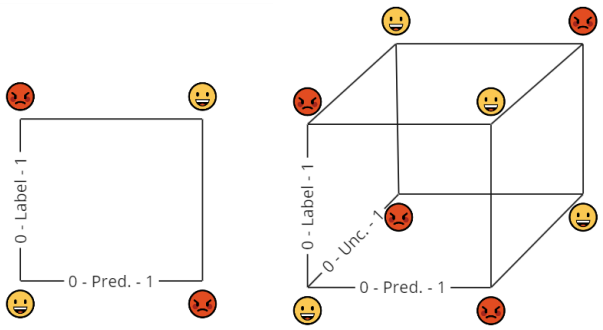
**Uncertainty-aware objectives.** One of the approaches to uncertainty estimation of the regression models is the *heteroscedastic* regression that takes both the variable mean and variance into account [18, 20]. So, the model trains to predict means and variances, and the uncertainty of the model predictions can be estimated using the variance values. Fortunately, classification models can also use a *squared error* (SE) loss. Hui and Belkin [10] demonstrated that the SE and CE-based computer vision models are close in accuracy. However, a SE loss needs some more training epochs. Kendall and Gal [11] dealt with two types of uncertainty, that are aleatoric (data uncertainty) and epistemic (model uncertainty), and proposed two approaches in uncertainty estimation. Kendall and Gal [11] declared that out-of-data examples cannot be identified with aleatoric uncertainty. The authors also proposed an approach that combines aleatoric and epistemic uncertainties. Further work by van Amersfoort et al. [22] deals with the deterministic uncertainty quantification method. The proposed model learns the positions of centroids of classes and trains kernels to estimate the distance between an input sample and centroids, which allows the inference model to recognize an out-of-data sample as uncertain. Sensoy et al. [21] developed a theory of evidence perspective and represented the model predictions as a Dirichlet density distribution over the softmax outputs and proposed a novel loss function. Collier et al. [3] proposed a method for training deep classifiers under heteroscedastic label noise. The method deals with the softmax temperature tuning that allows to control a bias-variance trade-off.

**Ensembling, test-time augmentation, and label smoothing.** Ashukha et al. [1] demonstrated that many ensembling techniques are equivalent to an ensemble of several independently trained networks in terms of test performance. Test-time augmentation is a technique that improves model performance using averaging the predictions [14]. Probably the simplest ways to make models be more robust to noise in labels are label smoothing [23] and data augmentation [19].

**Data uncertainty estimation in practice** Corrupted inputs [13] and corrupted labels [26], in-domain and out-of-domain distributions [15, 11, 3] are some of the poles of the research in the scope of data uncertainty estimation. The typical test of the models in practice is to use public datasets with corrupted (noisy) labels at the training and validation stages but with clean labels at the test stage [25, 26]. A number of methods try to detect input samples with incorrect labels and remove [3, 25, 26] or under-weight these samples [11, 4]. Han et al. [8] declared that models learn data with clean labels first and noisy labels then, and proposed a new paradigm called *co-teaching* with the training of two networks.

# 4 Methodology

In general, consider a model $\mathbf{f}[\mathbf{x}, \mathbf{w}]$ parameterized by weights $\mathbf{w}$ that maps an input $\mathbf{x}$ into logits $\mathbf{z}$ first and then into the hypothesis $\mathbf{h}$ that approximates the ground truth $\mathbf{y}$. The negative log-likelihood minimization [18, 2, 6] allows the formalization of the following uncertainty-aware loss functions for fitting and classification problems using different types of distributions for the outputs of the models.




(a) (b)

Figure 2: Binary classification intuition with the BCE loss (a) and the proposed binary B-loss eq. (4) (b) with respect to the values of the model's outputs (uncertainties $u = 1 - c$, predictions $h$), and labels $y$.

## 4.1 B-loss

This paragraph presents a specific interpretation of a binary classification model based on minimizing the *uncertainty-aware negative log-likelihood with the Bernoulli distribution* (B-model, B-loss). The proposed model is trained to ensure that true predictions are made certain and false predictions if they occur, are made uncertain (see fig. 2). The binary classifier estimates the certainty value $c \in (0, 1)$, which is the primary task in the proposed formalization. Additionally, the classifier estimates and enhances the similarity $\delta$ between the hypothesis $\mathbf{h}$ and the ground truth $\mathbf{y}$, which constitutes the secondary task in the proposed formalization.

**Binary classification.** Consider an $i^{\text{th}}$ sample and a model with logits $z^{(i)} = \left[z^{(i)}_{pred}, z^{(i)}_{cert}\right]$ which correspond to a prediction $h^{(i)} = \sigma(z^{(i)}_{pred})$, and the certainty $c_i = \sigma(z^{(i)}_{cert})$ associated with the prediction, respectively. Then, compare the prediction $h^{(i)}$ and the given label $y^{(i)}$ using a scalar product metric $\delta_i = y^{(i)} h^{(i)}$, and map this metric as a pseudo-label of a binary uncertainty estimator into the parameters of a Bernoulli probability mass function [18]:

$$p_i = p(\delta_i | c_i) = \begin{cases} 1 - c_i & \text{if } \delta_i \to 0, \\ c_i & \text{if } \delta_i \to 1, \end{cases} \tag{1}$$

where $\delta_i \in (0, 1)$ is the smoothed pseudo-label that characterizes the similarity between the label and the prediction.

eq. (1) is a discrete probability distribution for a random variable that takes the value 0 with probability $1 - c_i$, which is an incorrect prediction that corresponds to an uncertainty of the prediction, and the value 1 with probability $c_i$, which is a right prediction that corresponds to a certainty of the prediction. The Bernoulli distribution has an equivalent power law form [18]:

$$p_i = c_i^{\delta_i} (1 - c_i)^{1 - \delta_i}. \tag{2}$$

For a roll-out of dataset of $m$ i.i.d. pairs $\{x^{(i)}, y^{(i)}\}$ associated with the outputs of the model $\{h^{(i)}, c_i\}$, the joint probability [2] for the given probability mass function eq. (2) takes the following form:

$$P(\delta_1, \ldots, \delta_m \mid c_1, \ldots, c_m) = \prod_{i=1}^{m} c_i^{\delta_i} (1 - c_i)^{1 - \delta_i}. \tag{3}$$

The negative logarithm of the joint probability eq. (3) represents the proposed uncertainty-aware B-loss for the binary classification:

$$\mathcal{L}_{\text{B}} = -\frac{1}{m} \sum_{i=1}^{m} \left[\delta_i \log c_i + (1 - \delta_i) \log(1 - c_i)\right]. \tag{4}$$

eq. (4) intuition is demonstrated in fig. 2. The B-loss can be generalized for the case of multiclass classification.

**Multiclass (N-classes) classification.** Consider an $i^{\text{th}}$ sample and a model with logits $\mathbf{z}^{(i)} = [\mathbf{z}^{(i)}_{pred}, z^{(i)}_{cert}]$ which correspond to a vector of prediction $\mathbf{h}^{(i)} = \text{softmax}(\mathbf{z}^{(i)}_{pred})$, $\mathbf{h}^{(i)} \in \mathcal{R}^N$, and the certainty $c_i = \sigma(z^{(i)}_{cert})$ associated with the prediction, relatively. Then, compare the prediction vector $\mathbf{h}^{(i)}$ and the given one-hot encoded label vector $\mathbf{y}^{(i)}$ using a scalar product terms $\delta^{(i)}_k = y^{(i)}_k h^{(i)}_k$, and map this metrics as pseudo-labels into a probability mass function:

$$p_i = \prod_{k=1}^{N} \left(\frac{c_i}{N}\right)^{\delta^{(i)}_k} \left(\frac{1 - c_i}{N}\right)^{1 - \delta^{(i)}_k}, \tag{5}$$

where $\delta_k^{(i)} \in (0, 1)$ is the smoothed one-hot encoded pseudo-label that characterizes the similarity between the $k^{\text{th}}$ components of the label and the prediction vectors, $N$ is the number of classes.

Following the logical sequence given in section 4.1 and in [18], the joint probability for eq. (5) can be obtained, and then transformed into the negative log-likelihood (NNL):

$$NLL = -\frac{1}{m}\sum_{i=1}^{m}\left(\cos(\mathbf{h}^i, \mathbf{y}^i)\log\left(\frac{c^{(i)}}{N}\right) + (N-1)\left(1-\cos(\mathbf{h}^i, \mathbf{y}^i)\right)\log\left(\frac{1-c^{(i)}}{N}\right)\right), \tag{6}$$

where $\cos(\mathbf{h}^i, \mathbf{y}^i)$ is the smoothed pseudo-label that characterizes the cosine similarity between two N-dimensional vectors: the vector of prediction and the one-hot encoded label vector.

Finally, the proposed uncertainty-aware B-loss for the N-classes classification is the Kulback-Loeberg divergence between two distributions: the one-hot encoded smoothed pseudo-labels distribution and the NNL distribution eq. (6):

$$\mathcal{L}_{\text{B}} = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{N}\delta_k^{(i)}\log\delta_k^{(i)} + NLL. \tag{7}$$

where $m$ is the number of samples (in a batch), $N$ is the number of classes, $\delta_i = y^{(i)}h^{(i)}$ are the terms of a scalar product of the one-hot encoded label vector and the vector of prediction of the model, $c^{(i)}$ is the certainty of the prediction (fig. 4b).

## 4.2 N-loss

Since the binary classification can be assumed as a particular case of a multiclass classification, this section skips the binary classification paragraph.

**Multiclass classification.** Consider an $i$-th sample and a model with logits $\mathbf{z}^{(i)} = [\mathbf{z}_{mean}^{(i)}, z_{var}^{(i)}]$ which maps into the parameters of a multivariate normal distribution: the hypothesis or mean $\mathbf{h}^{(i)} = \mathbf{z}_{mean}^{(i)}$ that approximates the ground truth $\mathbf{y}^{(i)}$, and the variance $\sigma_{(i)}^2 = \exp(z_{var}^{(i)})$ that characterizes the uncertainty of the hypothesis, $f[\mathbf{x}^{(i)}, \mathbf{w}] = [\mathbf{h}^{(i)}, \sigma_{(i)}^2]$. In other words, it is assumed that the conditional probability distribution $p = p(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) = p(\mathbf{y}^{(i)}|\mathbf{f}[\mathbf{x}^{(i)}, \mathbf{w}])$ has the form of a multivariate normal distribution characterized by equal variances (spherical covariances) in N-dimensional space [17]:

$$p^{(i)} = \frac{\exp\left(-\frac{\sum_{k=1}^{N}\left(y_k^{(i)} - h_k^{(i)}\right)^2}{2\sigma_{(i)}^2}\right)}{\left(2\pi\sigma_{(i)}^2\right)^{\frac{N}{2}}}, \tag{8}$$

The multivariate normal distribution of (8) can be applied to the negative log-likelihood criterion of an uncertainty-aware negative log-likelihood loss (N-loss) for the regression [18]:

$$\mathcal{L}_N = \frac{1}{2m}\sum_{i=1}^{m}\left(\sum_{k=1}^{N}\frac{\left(y_k^{(i)} - h_k^{(i)}\right)^2}{\sigma_{(i)}^2} + N\left(s^{(i)} + r\right)\right), \tag{9}$$

where $m$ is the number of samples (in a batch), $\mathbf{y}^{(\text{i})}$, $s^{(i)} = \log\sigma_{(i)}^2$ is the log-variance, $r = \log 2\pi$ is the constant value.

The last term in (9) represents a constant that can be neglected. Kendall et al. [11] recommended to train the models to predict log-variances $s^{(i)} = \log\sigma_{(i)}^2$, because it is more numerically stable than the variance $\sigma_{(i)}^2$ and the loss avoids a potential division by zero:

$$\mathcal{L}_N = \frac{1}{m}\sum_{i=1}^{m}\left(e^{-s^{(i)}}\sum_{k=1}^{N}\left(y_k^{(i)} - h_k^{(i)}\right)^2 + Ns^{(i)}\right). \tag{10}$$

Thus, (10) represents a heteroscedastic regression loss [18, 11], generalized for the case of a space of N dimensions. Our proposal is to use this loss for classification problems.

The baseline loss in a multi-class classification problem is the cross-entropy loss [6, 18, 2]:

$$L_{CE} = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{N}y_k^{(i)}\log h_k^{(i)}. \tag{11}$$

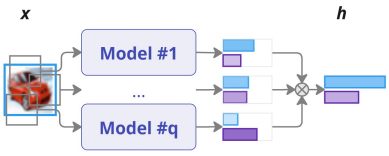



Figure 3: Ensemble of $q$ models makes the prediction $\mathbf{h}$ for the $q$ augmented copies of an input sample $\mathbf{x}$. Two-classes classification is demonstrated.

## 4.3 Ensembling

A set of $q$ models $\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_q$ with different random initialization of the weights are trained with the same dataset. This aggregation reduces overfitting and provides more robust estimates by averaging out individual model errors [1]. Each $j^{\text{th}}$ model predicts a class index for the given $i^{\text{th}}$ input:

$$\hat{y}^{(i,j)} = \arg\max_k h_k^{(i,j)}, \tag{12}$$

where $i, j, k$ are the dummy indexes that refer to $j^{\text{th}}$ augmented version of $i^{\text{th}}$ sample, and $k^{\text{th}}$ component of the prediction vector or class index, $i \in (1, m), j \in (1, q), k \in (0, N-1)$.

The final ensemble prediction class is typically determined by *majority voting* based on the class predictions of the individual $j^{\text{th}}$ model.

$$\hat{y}^{(i)} = \text{mode}\left(\hat{y}^{(i,1)}, \hat{y}^{(i,2)}, \ldots, \hat{y}^{(i,q)}\right). \tag{13}$$

The final ensemble prediction class can also be determined by *confidence-based weighted predictions* (see fig. 3). Each model predicts a class $\hat{y}^{(i,j)}$ and provides a confidence value $co^{(i,j)}$ for its prediction:

$$co^{(i,j)} = \max_k(h_k^{(i,j)}). \tag{14}$$

The aggregated confidence for class $k$ is:

$$co_k^{(i)} = \sum_{j=1}^{q} co^{(i,j)} \cdot \mathbb{I}\left(\hat{y}^{(i,j)} = k\right), \tag{15}$$

where $\mathbb{I}(\cdot)$ is the indicator function, which returns 1 if the condition is true and 0 otherwise, $co_k^{(i)}$ is the total confidence for class $k$ across all models.

The final predicted class is:

$$\hat{y}^{(i)} = \arg\max_k co_k^{(i)}. \tag{16}$$

The uncertainty estimation in deep ensembling is derived from the variance of the individual model predictions. Higher variance among the models' outputs indicates greater uncertainty, providing a measure of epistemic uncertainty.

## 4.4 Metrics

The standard classification metrics for the balanced datasets are the accuracy, receiver operating characteristic - area under curve (ROC-AUC) [2, 6]. A set of more specific metrics used in uncertainty quantification involves estimation the confidence (see eq. (14)) [15]: the Bries score, the entropy, the expected calibration error (ECE), the negative log-likelihood (NLL), the prediction interval coverage probability (PICP), the sharpness, etc. [16, 7, 5, 9].

Since the proposed B-model (see eq. (7)) and N-model (see eq. (10)) have an extra output, the following additional *certainty* metrics can be met:

- $c^{(i)} \in (0, 1)$ for the B-model ;
- $1 - sigm(s^{(i)}) \in (0, 1)$ for the N-model.

Both of the above metrics can be used as weights in eq. (16), thus the *certainty-based weighted predictions* should be met.

# 5 Results and Discussion

CIFAR-10N dataset [24] was split into training, validation, and test sets in the amounts of $[45000, 5000, 10000]$ samples, respectively. The models were trained with a 9-layer convolutional neural network [8, 25]. The network has 4.4 million parameters that were randomly initialized during training. The CNN architecture and most of the settings correspond to the experiments by Xia et al. [25] with minor changes: the models were trained for 20 epochs (200 in the original paper) using the Adam optimizer with a momentum of 0.9 and a batch size of 128, and a constant learning rate of 0.001 (in the original paper, the initial learning rate linearly decreased to zero starting from the $80^{\text{th}}$ epoch); image samples were transformed into tensors and normalized with means of $[0.491, 0.482, 0.447]$ and standard deviations of $[0.247, 0.243, 0.261]$.

Some of the experimental settings might differ from those of Xia et al. [25]: the model ensembling technique was applied; a random resized crop with a scale range of $[0.8, 0.1]$ and an aspect ratio range of $[0.9, 1.1]$ was applied to all samples in all sets as a transformation, so test set sampling was implemented using test-time augmentation with a random resized crop; models with the lowest validation loss were used for inference.

All the experiments were performed seven times with random seeds of $[42, 0, 17, 9, 3, 16, 2]$. The mean and standard deviation of experimental results were then reported. The multiple predictions obtained through model ensembling allowed for the calculation of final predictions using majority voting.

Accuracy was used [12] as metric. The obtained results were not compared with the state-of-the-art results. The latter aggregate multiple techniques and complex network architectures. So, the comparison would be unfair.

In this study, I trained a total of 63 distinct models across three architectures (CNN, ResNet50, and ViT[1]) using seven different weight initialization seeds for each combination of architecture and loss function. The models were trained on the CIFAR-10N dataset, which includes both clean and noisy labels, and were evaluated using three metrics: the mean accuracy of individual models, the accuracy of the ensemble using majority voting (EMV), and the accuracy of the ensemble with weighted predictions (EWP). Additionally, I included a full ensemble that combines all loss functions (CE, B, and N) for each architecture to further explore the potential of ensemble methods. The total size of all trained models is approximately 9 GB, and I conducted 26 experiments to compute all metrics.

The CNN models achieved the highest individual and ensemble accuracies among the three architectures. The proposed N-loss outperformed both the baseline CE loss and the proposed B-loss on clean labels, achieving a mean accuracy of 85.89% and an ensemble accuracy of 89.0% with majority voting. For noisy labels, the B-loss demonstrated superior performance, with a mean accuracy of 72.64% and an ensemble accuracy of 76.41%. These results suggest that the N-loss is more robust to clean labels, while the B-loss is better suited for noisy labels.

For ResNet50, the proposed B-loss achieved the best performance on noisy labels, with a mean accuracy of 65.22% and an ensemble accuracy of 76.52% using weighted predictions. The full ensemble of ResNet50 models achieved the highest accuracy of 77.34%, demonstrating the effectiveness of combining multiple models. However, the N-loss underperformed on noisy labels, with a mean accuracy of 62.05%, indicating that it may not be as effective for larger architectures like ResNet50.

The ViT models showed competitive performance, with the proposed B-loss and N-loss achieving similar results on noisy labels. The B-loss achieved a mean accuracy of 72.05%, while the N-loss achieved 71.40%. The ensemble with weighted predictions (EWP) further improved the accuracy, with the N-loss achieving 75.75% and the full ensemble reaching 79.98%. This highlights the potential of transformer-based architectures like ViT for handling noisy labels.

Across all architectures, the ensemble methods (EMV and EWP) consistently outperformed the mean accuracy of individual models. The weighted prediction ensemble (EWP) provided a slight improvement over majority voting (EMV) in most cases, particularly for ResNet50 and ViT. This demonstrates the value of incorporating confidence-based weighting in ensemble methods.

Due to computational constraints, experiments with label smoothing (LS) and full ensemble evaluations for the CNN architecture were not conducted. Future work could explore the impact of label smoothing on model performance, particularly for noisy labels.



(a) CE-model (b) B-model (c) N-model

Figure 4: Validation loss values during the training of the ensemble models in 20 epochs with clean and noisy data: CE-model (a), B-model (b), and N-model (c).

[1] https://huggingface.co/timm/vit_tiny_patch16_224.augreg_in21k

Table 1: Accuracy (%) of models trained on the CIFAR-10N dataset with clean and noisy labels, as well as with label smoothing (LS). Each model was trained using seven different weight initialization seeds for 20 epochs, then ensembled. The mean test accuracy of individual models is compared with the accuracy of the ensemble using majority voting (EMV) and the ensemble with weighted predictions (EWP). The best results are highlighted in bold.

| **Method** | **Arch.** | **#Param.** (train.par.) | **LS** | **Accuracy (single model / EMV / EWP),%** Clean | Noisy (worse) |
|---|---|---|---|---|---|
| Baseline CE loss | 9-l.CNN | 4.4 M (all) | 0.0 | $84.22 \pm 0.007/87.80/00.00$ | $71.36 \pm 0.004/74.26/00.00$ |
| Proposed B-loss | | | | $84.49 \pm 0.003/88.13/00.00$ | $\mathbf{72.64 \pm 0.004/76.41/00.00}$ |
| Proposed N-loss | | | | $\mathbf{85.89 \pm 0.005/89.0/00.00}$ | $70.13 \pm 0.007/74.90/00.00$ |
| Full ensemble | | | | $-/-/-$ | - |
| CE loss | ResNet50 | 25.6 M | 0.0 | $-/-/-/-/-$ | $64.89 \pm 0.004/69.24/00.00$ |
| Proposed B-loss | | | | $-/-/-/-/-$ | $\mathbf{65.22 \pm 0.009/69.32/76.52}$ |
| Proposed N-loss | | | | $-/-/-/-/-$ | $62.05 \pm 0.098/62.3/68.33$ |
| Full ensemble | | | | $-/-/-$ | 00.00/77.20/ 77.34 |
| CE loss | ViT | 5.7 M | 0.0 | $-/-/-/-/-$ | $70.40 \pm 0.028/73.11/00.00$ |
| Proposed B-loss | | | | $-/-/-/-/-$ | $72.05 \pm 0.013/74.31/74.04$ |
| Proposed N-loss | | | | $-/-/-/-/-$ | $\mathbf{71.40 \pm 0.034/75.26/76.75}$ |
| Full ensemble | | | | $-/-/-$ | 00.00/79.75/ 79.98 |

# 6 Ethics

In this work, I utilized Large Language Models (LLMs), specifically DeepSeek and Perplexity, to assist with certain aspects of the project. Below, I outline how these tools were used and reflect on their contributions relative to our own efforts.

DeepSeek was used to assist in writing, debugging, and optimizing parts of the code. For example, it helped generate boilerplate code for model evaluation and provided suggestions for improving the efficiency of certain functions. However, all code was carefully reviewed, tested, and adapted by the authors to ensure correctness and alignment with the project's goals.

Both DeepSeek and Perplexity were used to assist in drafting, rephrasing, and improving the clarity of the text. This included refining technical explanations, improving the flow of the document, and ensuring consistent terminology. While these tools provided valuable assistance, all content was critically reviewed and edited by me to maintain accuracy and coherence.

# 7 Conclusion

In this work, I explored the effectiveness of different loss functions (CE, B, and N) and ensemble methods (majority voting and confidence-based weighting) for training models on the CIFAR-10N dataset, which includes both clean and noisy labels. My experiments, involving 63 distinct models across three architectures (CNN, ResNet50, and ViT), revealed several key insights. First, the proposed N-loss demonstrated superior performance on clean labels, while the B-loss excelled on noisy labels, highlighting the importance of tailoring loss functions to the nature of the data. Second, ensemble methods consistently improved model performance, with confidence-based weighting (EWP) providing a slight but meaningful advantage over majority voting (EMV) in most cases. Finally, the full ensemble, which combined all architectures and loss functions, achieved the highest accuracy of 79.98% on noisy labels, underscoring the power of diversity in model ensembles.

Beyond these results, my work raises important questions about the trade-offs between model complexity, computational cost, and performance. For instance, while larger architectures like ResNet50 and ViT showed competitive results, their computational demands may limit their practicality in resource-constrained settings. Conversely, simpler models like the CNN achieved strong performance with significantly fewer parameters, suggesting that careful architecture selection is crucial for balancing efficiency and accuracy. Additionally, the success of confidence-based weighting in improving ensemble performance opens new avenues for research into more sophisticated aggregation methods, particularly for noisy datasets.

By systematically evaluating these approaches, I not only provide a clearer understanding of how different loss functions and ensemble strategies interact with clean and noisy labels but also offer practical guidance for researchers and practitioners working on real-world datasets where label noise is inevitable. While my work does not claim to solve all challenges associated with noisy labels, it contributes to making the problem clearer and more

approachable, paving the way for future innovations in robust machine learning.

## References


[1] A. Ashukha, A. Lyzhov, D. Molchanov, and D. Vetrov. Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. *arXiv preprint arXiv:2002.06470*, 2020.

[2] C. M. Bishop and N. M. Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[3] M. Collier, B. Mustafa, E. Kokiopoulou, R. Jenatton, and J. Berent. A simple probabilistic method for deep classification under input-dependent label noise. *arXiv preprint arXiv:2003.06778*, 2020.

[4] E. Englesson, A. Mehrpanah, and H. Azizpour. Logistic-normal likelihoods for heteroscedastic label noise, 2023.

[5] T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[6] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[7] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. *International Conference on Machine Learning*, pages 1321–1330, 2017.

[8] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels, 2018.

[9] F. Hernandez, L. Bertino, G. Brassington, E. Chassignet, J. Cummings, F. Davidson, M. Drevillon, G. Garric, M. Kamachi, J. M. Lellouche, et al. Probabilistic forecasting in meteorology: A review. *Quarterly Journal of the Royal Meteorological Society*, 141(688):318–350, 2015.

[10] L. Hui and M. Belkin. Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks, 2021.

[11] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[12] A. Kumar, P. Liang, and T. Ma. Verified uncertainty calibration, 2020.

[13] E. Mintun, A. Kirillov, and S. Xie. On interaction between augmentations and corruptions in natural corruption robustness, 2021.

[14] D. Molchanov, A. Lyzhov, Y. Molchanova, A. Ashukha, and D. Vetrov. Greedy policy search: A simple baseline for learnable test-time augmentation, 2020.

[15] T. Pearce, A. Brintrup, and J. Zhu. Understanding softmax confidence and uncertainty. *CoRR*, abs/2106.04972, 2021. URL `https://arxiv.org/abs/2106.04972`.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] S. Prince. *Computer Vision: Models Learning and Inference*. Cambridge University Press, 2012.

[18] S. J. Prince. *Understanding Deep Learning*. MIT Press, 2023. URL `http://udlbook.com`.

[19] S.-A. Rebuffi, S. Gowal, D. A. Calian, F. Stimberg, O. Wiles, and T. A. Mann. Data augmentation can improve robustness. *Advances in Neural Information Processing Systems*, 34:29935–29948, 2021.

[20] M. Seitzer, A. Tavakoli, D. Antic, and G. Martius. On the pitfalls of heteroscedastic uncertainty estimation with probabilistic neural networks. *arXiv preprint arXiv:2203.09168*, 2022.

[21] M. Sensoy, L. Kaplan, and M. Kandemir. Evidential deep learning to quantify classification uncertainty, 2018.

[22] J. van Amersfoort, L. Smith, Y. W. Teh, and Y. Gal. Simple and scalable epistemic uncertainty estimation using a single deep deterministic neural network. *CoRR*, abs/2003.02037, 2020. URL `https://arxiv.org/abs/2003.02037`.

[23] J. Wei, H. Liu, T. Liu, G. Niu, M. Sugiyama, and Y. Liu. To smooth or not? when label smoothing meets noisy labels. *arXiv preprint arXiv:2106.04149*, 2021.

[24] J. Wei, Z. Zhu, H. Cheng, T. Liu, G. Niu, and Y. Liu. Learning with noisy labels revisited: A study using real-world human annotations, 2022. URL `https://arxiv.org/abs/2110.12088`.

[25] X. Xia, T. Liu, B. Han, M. Gong, J. Yu, G. Niu, and M. Sugiyama. Sample selection with uncertainty of losses for learning with noisy labels, 2021.

[26] Q. Yao, H. Yang, B. Han, G. Niu, and J. Kwok. Searching to exploit memorization effect in learning from corrupted labels, 2020.