Computer Vision with Real-World Data

Student	Ilia Milioshin
Student's time capacity	8 ± 3 hours
Simulation time capacity (GPU: 20 GB VRAM)	10 ± 5 hours

General Instruction

Please use Russian or English language when preparing this document, but not their mixture. The exception is for the code in the Appendix section which may be a mixture of operators in English and comments in Russian.

Students are welcome to use any tool that is suitable for preparing high-quality work. However, we ask students to keep in mind two important criteria. First, we expect students to fully describe their methodology, and any tool that is important to that methodology, including the use of LLMs, should be described also. For example, students should mention tools (including LLMs) that were used for data processing or filtering, visualization, facilitating or running experiments, reviewing, coding, translating, proving theorems, etc. It may also be advisable to describe the use of LLMs in implementing the method (if this corresponds to an important, original, or non-standard component of the approach). Second, students are responsible for the entire content of their work, including all text and figures, so while students are welcome to use any tool they wish for writing, they must ensure that all text is correct and original.

A few practices are strongly discouraged: plagiarism, falsification, and any other methods that divert students' internal resources and abilities away from truly immersing themselves in solving this assignment.

So, the goal should not be to avoid challenges or merely achieve formal results, such as grades, but to engage deeply with the material and develop meaningful solutions.

Goals

The primary goals of this assignment are for students to:

1. Learn how to work with real-world data that may contain human labeling errors.
2. Understand how to design AI systems that account for uncertainty and doubt.
3. Explore methods and model configurations to enhance robustness when dealing with imperfect data.

To achieve these goals, students will compare three types of loss functions in the context of a classification problem: the *cross entropy loss* (CE loss), as a standard loss function for classification tasks; a *custom loss*, called B-loss (ideally, adapted or created by the student) that can estimate the uncertainty of the model's predictions; the *Heteroscedastic Regression Loss based on a multivariate normal distribution* (N-loss) which is normally used in fitting problems and which capable of estimating the uncertainty of the model's predictions in classification.

Through this comparison, students will gain insights into how different loss functions handle imperfect data and how to make models more resilient to such challenges.

Tasks and Requirements

The following points should be met:

1. Fill your name in the form above
2. Review the dataset CIFAR-10N and related work
3. Check the proposed methodology on loss functions, ensembling and metrics
4. Implement the proposed methodology in code for training an ensemble with a given loss and a list of seeds
5. Implement the label smoothing teqniques and alternative ways to calculate final ensemble predictions using majority voting and weighted predictions (check your personal task with an instructor)

6. Perform a computational experiment, fill the table and figure of the results

7. Fill in the remaining sections of the report, including Related work, Results and discussion, Conclusion, and Introduction

8. Submit the code and report

Bonus (complete one of the following points as a bonus task):

1. Revise the proposed B-model or N-model, enhance it and obtain higher metrics in additional experiments

2. Implement a more advance ANN, e.g. using recommendations from papers related to CIFAR-10N dataset

3. Achieve the original results with accuracy of top 10 solutions in accordance with SOTA rating for CIFAR-10N

Sections that require revision and completion: 1, 2, 4, 5, 6.

Evaluation Criteria

The work will be evaluated based on the following criteria:

1. **Basic Level (65–75%)**:
   - The main tasks have been completed.
   - Verifiable results have been obtained.
   - A well-formatted written report and the main code have been submitted.

2. **Intermediate Level (76–89%)**:
   - The approach to solving the problem and the results obtained demonstrate originality.
   - Additional efforts have been made to achieve better results.
   - The work is neatly presented and shows a deep understanding of the topic.

3. **Advanced Level (90–100%)**:
   - One of the *bonus track* questions has been solved, or results exceeding basic expectations have been achieved.
   - The work demonstrates exceptional quality, creativity, and independence.

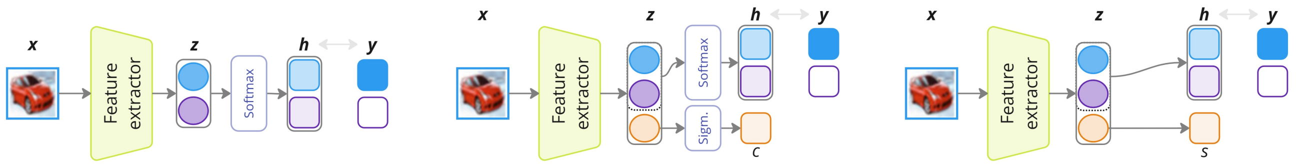
1 Introduction

The increasing prevalence of real-world datasets with human annotation errors poses significant challenges to the development of robust AI systems. Addressing this issue requires models capable of handling uncertainty and label noise without sacrificing accuracy. This work explores and compares three loss functions—Cross-Entropy (CE) loss, B-loss, N-loss, and UCA-loss — within the CIFAR-10N dataset, focusing on their ability to improve classification performance under noisy conditions. Special attention is given to uncertainty estimation and ensembling techniques to enhance model resilience.

2 Related work

Learning with Real-World Human Annotations. Rabbani et al. proposed a novel approach to handle noisy labels in the CIFAR-10N dataset by leveraging a noise-robust loss function combined with a label correction mechanism Rabbani and Bartoli [21]. Their method involves training a deep neural network with a modified loss function designed to account for label noise. This is achieved by employing auxiliary deep neural networks to estimate the certainty of the labeled data. The proposed methodology is model-independent and can be adapted to various machine learning tasks. For example, the UCA head architecture can be used for computing B-loss or N-loss. However, it requires hyperparameter tuning, which increases the overall complexity.

Another approach to handling uncertainty in the CIFAR-10N dataset is to mitigate the influence of noise through over-parameterization. This involves using models with significantly more parameters than the number of training



(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).

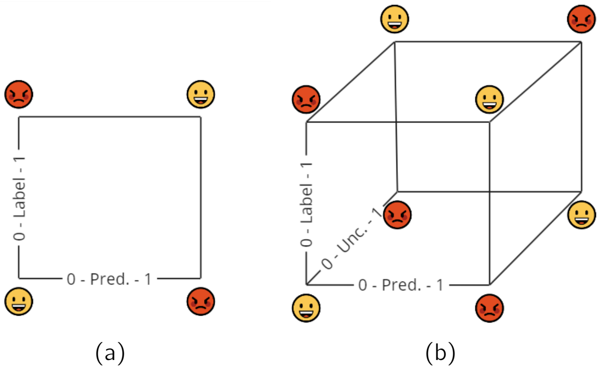
samples. Such over-parameterized models are better able to capture the underlying data distribution, even in the presence of noisy labels. Liu et al. [14] demonstrated that over-parameterized models can fit noisy data while still generalizing well to clean data. In the context of CIFAR-10N, the proposed methodology achieves state-of-the-art test accuracy against noisy labels. This is accomplished by leveraging the model's ability to memorize the training data, including noisy labels, while applying regularization techniques to prevent overfitting. N-loss also has an implicit regularization effect and could potentially be enhanced by combining it with sparse over-parameterization. Nevertheless, the suggested technique does not estimate confidence scores, unlike B-loss or N-loss.

Finally, soft labels can be utilized, as demonstrated by Kim et al. Kim et al. [12]. Instead of relying solely on model predictions, Learning with Structural Labels (LSL) derives labels from feature distributions using reverse k-NN. This method is more robust to outliers compared to standard k-NN. Moreover, the proposed methodology achieves state-of-the-art (SOTA) performance on CIFAR-10NW. Thus, reverse k-NN can be incorporated into the B-loss framework to better discriminate between uncertain and clean samples. However, similar to the aforementioned UCA, this method also requires hyperparameter tuning.

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 [20, 23]. 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. [25] 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. [24] 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 [16]. Probably the simplest ways to make models be more robust to noise in labels are label smoothing [26] and data augmentation [22].

Data uncertainty estimation in practice Corrupted inputs [15] and corrupted labels [29], in-domain and out-of-domain distributions [17, 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 [28, 29]. A number of methods try to detect input samples with incorrect labels and remove [3, 28, 29] 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.



(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.

3 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 [20, 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.

3.1 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 δ 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^{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 [20]:

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

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 [20]:

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

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:

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

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

Following the logical sequence given in section 3.1 and in [20], 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):

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

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).

3.2 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 $\mathbf{z}^{(i)} = [\mathbf{z}^{(i)}_{mean}, z^{(i)}_{var}]$ which maps into the parameters of a multivariate normal distribution: the hypothesis or mean $\mathbf{h}^{(i)} = \mathbf{z}^{(i)}_{mean}$ that approximates the ground truth $\mathbf{y}^{(i)}$, and the variance $\sigma^2_{(i)} = \exp\left(z^{(i)}_{var}\right)$ that characterizes the uncertainty of the hypothesis, $f[\mathbf{x}^{(i)}, \mathbf{w}] = [\mathbf{h}^{(i)}, \sigma^2_{(i)}]$. 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 [19]:

$$p^{(i)} = \frac{\exp\left(-\frac{\sum_{k=1}^{N}\left(y^{(i)}_k - h^{(i)}_k\right)^2}{2\sigma^2_{(i)}}\right)}{\left(2\pi\sigma^2_{(i)}\right)^{\frac{N}{2}}}, \tag{8}$$

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 [20]:

$$\mathcal{L}_N = \frac{1}{2m}\sum_{i=1}^{m}\left(\sum_{k=1}^{N}\frac{\left(y^{(i)}_k - h^{(i)}_k\right)^2}{\sigma^2_{(i)}} + N\left(s^{(i)} + r\right)\right), \tag{9}$$

where m is the number of samples (in a batch), $\mathbf{y}^{(\text{i})}$, $s^{(i)} = \log\sigma^2_{(i)}$ 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 $s^{(i)} = \log\sigma^2_{(i)}$, because it is more numerically stable than the variance $\sigma^2_{(i)}$ and the loss avoids a potential division by zero:

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

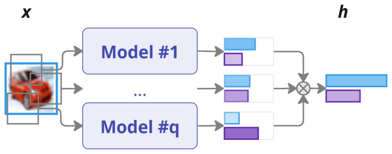


Figure 3: Ensemble of q models makes the prediction $\mathbf{h}$ for the q augmented copies of an input sample $\mathbf{x}$. Two-classes classification is demonstrated.

Thus, (10) represents a heteroscedastic regression loss [20, 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, 20, 2]:

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

3.3 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^{th} model predicts a class index for the given i^{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^{th} augmented version of i^{th} sample, and k^{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^{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.

3.4 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)) [17]: the Bries score, the entropy, the expected calibration error (ECE), the negative log-likelihood (NLL), the prediction interval coverage probability (PICP), the sharpness, etc. [18, 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.

4 Results and Discussion

CIFAR-10N dataset [27] 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, 28]. 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. [28] 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. [28]: 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 [13] 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.

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.

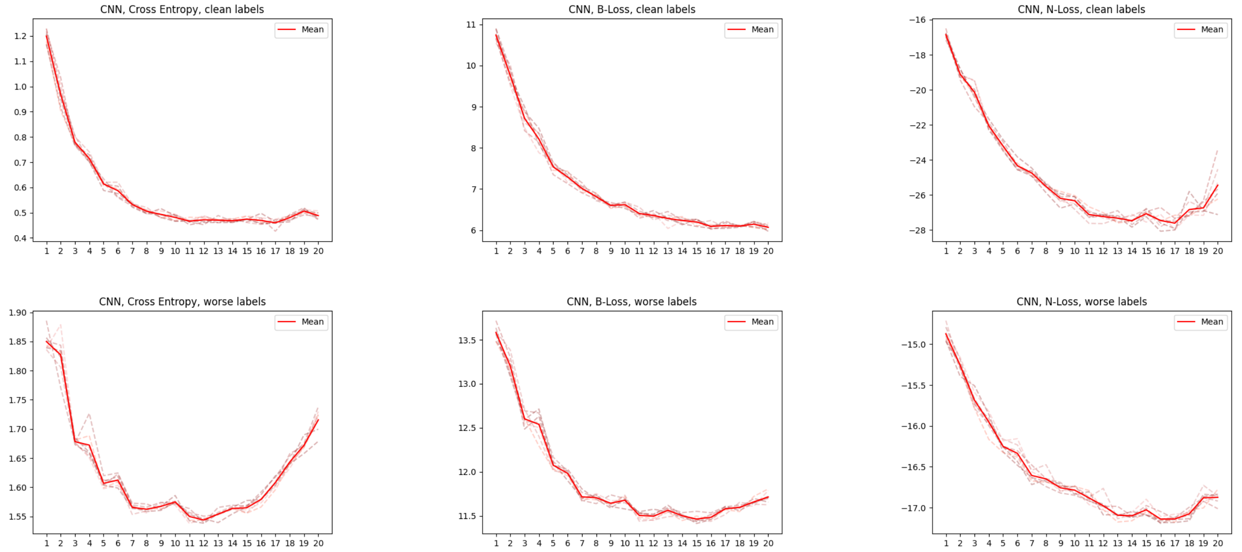
Method	**Arch.**	**#Param.**	**LS**	**Accuracy (single model / EMV / EWP),%**	
		(train.par.)		Clean	Noisy (worse)
Baseline CE loss	9-l.CNN	4.4 M (all)	0.0	$84.46 \pm 0.603/88.24/88.44$	$71.53 \pm 0.525/75.11/75.38$
Proposed B-loss				$84.15 \pm 0.407/87.90/87.90$	$72.68 \pm 0.435/76.38/76.67$
Proposed N-loss				$\mathbf{85.93 \pm 0.685/88.97/89.02}$	$73.08 \pm 0.416/76.42/76.67$
Proposed UCA-loss				- / - / -	$71.48 \pm 0.518/74.86/74.74$
Baseline CE loss	ResNet50	22.5 K (head only)	0.0	- / - / -	$72.40 \pm 0.371/72.47/69.49$
Proposed B-loss				- / - / -	$72.44 \pm 0.015/72.46/72.46$
Proposed N-loss				- / - / -	$\mathbf{74.48 \pm 0.060/74.50/74.50}$

5 Ethics

LLM utilization:

- Literature Review: LLMs were used to improve section readability. Each reviewed item was placed into a separate paragraph containing its key insights, applicability to a discussed topic, and downsides.
- Coding: Type hints were added by LLMs to improve code readability.
- Writing and Editing: LLMs slightly rewrote 1, 2, and 6 to keep uniform style.
- Critical Thinking: Figures and tables were pruned by LLMs to keep only relevant to study info.

Other aspects of this study were made without utilizing LLMs.



(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).

6 Conclusion

This study demonstrates that uncertainty-aware loss functions, such as B-loss, N-loss, and UCA-loss, offer a meaningful advantage over standard CE loss when handling noisy labels. Beyond improving accuracy, these methods enable the model to quantify prediction certainty, an essential capability in real-world noisy scenarios. However, enhancing model robustness requires balancing complexity and interpretability, particularly when incorporating advanced architectures or hyperparameter-sensitive techniques. Finally, uncertainty modeling not only increases classification reliability but also deepens our understanding of how AI systems process imperfect information.

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] N.-r. Kim, J.-S. Lee, and J.-H. Lee. Learning with structural labels for learning with noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 27610–27620, June 2024.

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

[14] S. Liu, Z. Zhu, Q. Qu, and C. You. Robust training under label noise by over-parameterization. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 14153–14172. PMLR, 17–23 Jul 2022. URL `https://proceedings.mlr.press/v162/liu22w.html`.

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

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

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

[18] 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.

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

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

[21] N. Rabbani and A. Bartoli. Unsupervised confidence approximation: Trustworthy learning from noisy labelled data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 4609–4617, October 2023.

[22] 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.

[23] 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.

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

[25] 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`.

[26] 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.

[27] 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`.

[28] 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.

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