# Detecting Noisy Labels Using Early Stopped Models

**Lea Hergert*** HERGERT@INF.U-SZEGED.HU
*University of Szeged, Hungary*

**Mark Jelasity** JELASITY@INF.U-SZEGED.HU
*University of Szeged, Hungary*
*HUN-REN SZTE Research Group on Artificial Intelligence, Szeged, Hungary*

**Editors:** Manuel Grana, Pawel Ksieniewicz, Leandro Minku, Pawel Zyblewski

## Abstract

We are concerned with the problem of identifying samples with noisy labels in a given dataset. Using the predictions of a well-generalizing model to flag incorrectly predicted labels as noisy is a known method but it is not considered competitive. At the same time, it has been observed recently that gradient descent fits clean samples first, and the noisy samples are memorized later. Inspired by related theoretical results, we revisit the idea of using the predictions of an early stopped model to classify samples as noisy or clean. We offer two key improvements that allow this strikingly simple approach to outperform some well-known methods. First, we use the model over its own training set to directly exploit the so-called clean priority learning phenomenon. Second, we use an ensemble of model checkpoints around the early stopping point to reduce the variance of the predictions. We also introduce a novel method that makes use of the same early stopped model ensemble, but classifies samples based on the per-sample gradient of the loss, motivated by recent theoretical results on clean priority learning. Our approaches only passively observe a normal training run and collect checkpoints. No extra input samples are added, no thresholds are tuned, and no pre-trained models are used. Through empirical evaluations, we demonstrate that our methods are competitive with other approaches from related work for both detecting noisy samples and for noise-filtered training.

**Keywords:** Limited label access

## 1. Introduction

We are concerned with the problem of identifying corrupted labels in the training dataset of a given supervised classification task (Northcutt et al., 2021; Pleiss et al., 2020). Solving this problem has several applications. For example, we can efficiently improve the labeling of noisy datasets if we can focus only on problematic samples, or we can improve the generalization of the trained model via removing noisy samples from the training set.

A naive and simple method to detect noisy samples is training a model with good generalization, and labeling those examples noisy that the model predicts incorrectly. Variants of this method (e.g., INCV (Chen et al., 2019)) are sometimes used as a baseline, but in general this approach is *not considered state-of-the-art* (Northcutt et al., 2021; Pleiss et al., 2020).

At the same time, results from other areas strongly suggest that well-configured *early-stopped training in itself is capable of filtering noise.* This was shown both theoretically (Liu

* Corresponding author.

et al., 2023; Li et al., 2020) and empirically (Arpit et al., 2017; Dong et al., 2022). In these works, the common observation is that in the early phases of training, gradient descent mostly ignores the noisy samples, and memorization begins only in later stages eventually resulting in overfitting. This has been observed even in the presence of large amounts of label noise (Arpit et al., 2017; Dong et al., 2022).

*These observations raise the question whether, if done right, early stopped models can be used to identify noisy samples after all.* A positive answer would be very desirable, because that way, the training algorithm would require little or no modification and we could offer a noise filtering approach that is very simple, natural, and efficient.

In order to answer this question, we revisit the naive method of using early stopped models. We argue that they are indeed capable of predicting noisy samples well, even outperforming some well-known baselines, but this requires attention to a number of details:

- The early stopped model needs to be used on its own training set, as opposed to an independent set. This is acceptable methodologically because the task at hand does not require us to clean independently sampled sets, the target is the training set.
- We need to use an ensemble of a small number of model checkpoints close to the early stopping point. This helps us reduce the variance of the predictions, thereby increasing accuracy.
- We need to consider the learning rate schedule of gradient descent carefully, because it affects not only the generalization of the early stopped model but also its usability for noise prediction.

Apart from using the prediction of the early stopped model ensemble directly, we also introduce a more sophisticated approach directly inspired by the theoretical discussion of Liu et al. (Liu et al., 2023). Instead of the prediction over a sample, this method is based on the gradient of the loss over the given sample according to the bias parameters of the feature layer (the layer before the logit layer) assuming the architecture under investigation has such parameters. In particular, we use the observation that the sum of the partial derivatives of these parameters is expected to be negative for clean samples and positive for noisy samples.

We conduct an empirical investigation where we explore the potential effects of learning rate schedules and ensemble types, and we also present a comparison with related work over the tasks of noisy label identification and noise-filtered training. We argue that the proposed approach is competitive with the state-of-the-art, even with models that rely on external information such as foundation models in the given domain.

### 1.1. Our Contributions

- We revisit the simple and natural method of using an early stopped model to identify noisy labels in a classification dataset, and we propose a number of improvements that make it competitive with state-of-the-art noise detectors.
- Inspired by theoretical results by Liu et al. (Liu et al., 2023), we improve the performance of the early stopped models on certain identification tasks by using a novel metric based on the per-sample gradient of the loss.

- Based on our empirical evaluation, we argue that our methods are competitive on the identification task and the filtered training task.

### 1.2. Related Work

Here, we focus only on those works that tackle the noisy label detection problem, where the goal is to identify which training samples have corrupted labels in a given dataset.

There are learning-based methods that require training a model. For example, some methods train multiple models simultaneously to filter out noisy samples, like Co-teaching (Han et al., 2018) or MentorNet (Jiang et al., 2018). INCV (Chen et al., 2019) uses multiple iterations of cross-validation to randomly split the dataset and removes examples with large loss values. CORES$^2$ (Cheng et al., 2021) sieves out corrupted samples with the introduction of a confidence regularizer. Confident Learning (Northcutt et al., 2021) selects samples with corrupted labels by ranking samples based on the classification probabilities of a trained model.

Observing training dynamics is also a commonly used strategy. O2U-net (Huang et al., 2019) uses a model that cycles between over- and underfitting and also removes samples with high average loss values. AUM (Pleiss et al., 2020) uses the difference between the logit value of a sample's assigned class and the highest logit value from the non-assigned classes to identify noisy samples. The method of (Jia et al., 2023) trains a model on a dataset with synthetic noise and then trains an LSTM network for noisy sample detection on the training dynamics of the first model.

Another approach proposed by (Yu et al., 2023) assumes the availability of a small amount of reliably clean data that can help find noisy samples by formulating the task as a multiple hypothesis testing problem.

Recently a more data-centric way was proposed with SimiFeat (Zhu et al., 2022) that uses a pre-trained model for feature extraction and assumes that samples with similar features are more likely to share the same clean label.

In our approach, we do not rely on any external information apart from observing a normal training run.

## 2. The Problem of Detecting Noisy Labels

We focus on the multi-class classification problem where the task is to find the parameters $\theta$ of a feed-forward neural network $f(\cdot;\theta) : \mathcal{X} \to [0,1]^C$, where $C$ is the number of classes, and the value of $f(x;\theta)$ is the predicted probability distribution over the classes, given $x$.

We are given a dataset $\mathcal{D} = \{(x_i, y_i) | i = 1, \ldots, n\}$ where $x_i \in \mathcal{X}$ and $y_i \in \{0,1\}^C$ is the one-hot encoded hard label. We assume that $f$ is trained via solving the optimization problem

$$\arg\min_{\theta} \sum_{(x,y)\in\mathcal{D}} \ell(y, f(x;\theta)), \tag{1}$$

where $\ell(y, f(x;\theta))$ is the loss function.

Ideally, all the $(x, y) \in \mathcal{D}$ samples are drawn i.i.d. from an underlying noise-free data distribution $p(x, y)$. However, in practice, datasets contain noise, which we model by

assuming a noise distribution $p_{noisy} \neq p$. Following general practice, we assume that $x$ is drawn from the true marginal $p(x) = \sum_y p(x, y)$ and only the label $y$ is noisy, that is,

$$p_{noisy}(x, y) = p_{noisy}(y|x) \sum_y p(x, y). \tag{2}$$

Dataset $\mathcal{D}$ is modeled as the union $\mathcal{D} = \mathcal{D}_{clean} \cup \mathcal{D}_{noisy}$ where $\mathcal{D}_{clean}$ and $\mathcal{D}_{noisy}$ are disjunct, and they contain samples drawn from $p(x, y)$ and $p_{noisy}(x, y)$, respectively.

*The task of noisy label detection is to classify a given sample $(x, y) \in \mathcal{D}$ as noisy or clean.*

## 3. An Overview of Our Approach

As discussed before, our goal is to investigate whether the simple and natural approach of using early stopped model checkpoints from a 'vanilla' training run can provide state-of-the-art performance in detecting noisy labels. We can break this approach down into the following three steps:

**Training:** Run a training algorithm on a noisy dataset $\mathcal{D}$ to find the optimal parameters $\theta^*$ of the model $f(\cdot; \theta)$

**Checkpoint Selection:** Select $k$ model checkpoints $\theta_1, \ldots, \theta_k$ from the training run

**Noise Detection:** Use all the $k$ model checkpoints to classify each training sample as noisy or clean, and output an ensemble decision for each sample

There are many details that need to be clarified and investigated. For example, what is the role of the training algorithm hyperparameters, most importantly, the learning rate schedule? How should we select the best $k$ checkpoints? How exactly should we use a given model checkpoint to classify a given sample as noisy or clean?

In the following sections, we discuss these issues. We start with the noise detection methods in Section 4, where we present two possible ways of using a single, well-generalizing model checkpoint to classify a sample as noisy or clean. In Section 5 we move on to the problem of checkpoint selection, in the light of the applied training algorithm. Finally, Section 6.2 studies the effect of the different ensemble hyperparameters such as size and aggregation method.

## 4. Noise Detection

Here, we assume that we are given a model checkpoint $\theta$ and our task is to propose a method for using the model $f(\cdot; \theta)$ to predict whether a given sample $(x, y)$ has a noisy label or not.

### 4.1. Prediction-based Method

Perhaps the simplest method for solving this task is to classify $(x, y)$ as noisy if and only if $f(x; \theta) \neq y$. Our main contribution is that we show empirically that this simple and natural method is surprisingly competitive when model selection is performed well.

### 4.2. Bias Gradient Sum Sign (BGSS) Method

Here, we are directly inspired by ideas from Liu et al. (2023) where it was argued that the per-sample gradients over noisy and clean samples tend to point in the opposite direction in the case of uniform label noise during the early phases of learning, called clean priority learning.

Like the prediction-based method, this method can be used without any parameter tuning, as we explain later. The only difference is that, instead of the prediction of the model, we will use a different function of the input sample and the model.

In the following, we will introduce this function that we shall call the bias gradient sum, or BGS for short. We will argue that for a sample $(x, y)$ and model parameter $\theta$ the sign of $BGS(x, y; \theta)$ tends to be positive if the sample is noisy and negative if it is clean. We shall call the BGS-based noise detection algorithm BGSS that stands for BGS Sign. Let us now go into the details of the BGSS method.

#### 4.2.1. Assumptions on the Network Architecture for BGSS

The architecture of $f$ is assumed to follow the general practice in deep learning, namely we assume that the output is computed using the softmax function over the vector of logits, and the logits are computed using a fully connected layer from the feature vector. That is,

$$f(x) = \text{softmax}(z(x)), \text{ and } z(x) = \theta_z g(x) + \theta_{zb}, \tag{3}$$

where $z(x)$ is the logit vector, $g(x)$ is the feature vector, $\theta_z$ is the weight matrix of the logit layer, and $\theta_{zb}$ is the bias vector of the logit layer. Furthermore, we assume that $g(x)$ is given by

$$g(x) = \max(0, (G(x) + \theta_{gb})). \tag{4}$$

In other words, $g(x)$ is assumed to have a bias vector $\theta_{gb}$ as a parameter and ReLU activation is applied. We do not make any further assumptions about earlier layers contained in $G(x)$. Note that pooling is often performed as well after the ReLU activation. For the sake of simplicity of notation, we ignore the pooling layer here because it does not change the discussion in any significant way (but we do not ignore it in our implementation).

#### 4.2.2. The Sign of the Bias Gradient Sum

Our detection method is based on the gradient of the loss according to the feature layer bias $\theta_{gb}$ in Equation (4). More precisely, we compute the sum of the partial derivative values over the feature bias parameters, and we take the sign of this sum.

Let us introduce a name for this quantity for easier reference. For a sample $(x, y)$,

$$BGS(x, y) = \sum_j \frac{\partial \ell(y, f(x; \theta))}{\partial \theta_{gb,j}}, \tag{5}$$

where BGS stands for bias gradient sum.

*We empirically find that the clean samples have negative BGS and the noisy samples have positive BGS on average extremely reliably.* In Figure 1, we illustrate the dynamics of the average BGS value for the clean and noisy examples in a number of settings. Note that

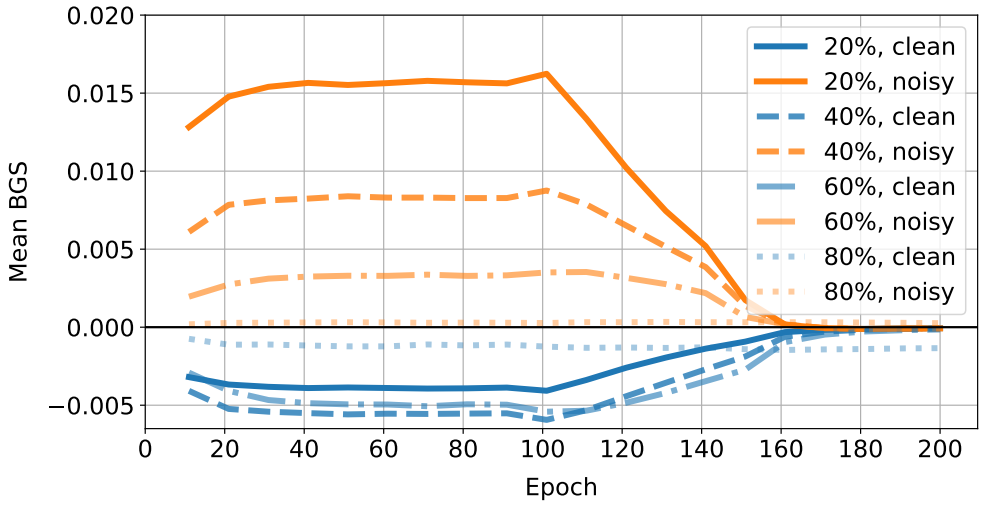


Figure 1: The average BGS of clean and noisy samples with uniform label noise with different noise rates during training a ResNet-18 network on CIFAR-10. The training setup is described in Section 6. The early stopping point is a few epochs after epoch 100 for all the noise rates.

the BGS curves begin to approach zero after the early stopping point, which is a few epochs after epoch 100 for all the noise rates.

Let us apply the approach used in (Liu et al., 2023) to understand this empirical observation better. The vector $\nabla_\theta \ell(y, f(x;\theta))$ is called the *per-sample gradient* of the loss with respect to the neural network parameters $\theta$. For a sample $(x, y)$, let $c(x)$ be the class label of $x$ such that $y_i = 1$ if $i = c(x)$, otherwise $y_i = 0$. Then, assuming cross-entropy loss, we have

$$\nabla_\theta \ell(y, f(x;\theta)) = \nabla_\theta - \log f(x;\theta)_{c(x)} = \sum_{i=1}^{C} (f(x;\theta)_i - y_i) \nabla_\theta z(x;\theta)_i. \tag{6}$$

This expression was also used in (Liu et al., 2023). Now, the partial derivative according to the bias of the $j$th feature $\theta_{gb,j}$ for a sample $(x, y)$ using Equations (3), (4) and (6) is

$$\frac{\partial \ell(y, f(x;\theta))}{\partial \theta_{gb,j}} = \sum_{i=1}^{C} (f(x;\theta)_i - y_i) \theta_{z,ij} 1_{g_j > 0}, \tag{7}$$

where $\theta_{z,ij}$ is the weight between the $i$th logit neuron $z(x;\theta)_i$, and the $j$th feature $g(x;\theta)_j$. Function $1_{g_j>0}$ is the indicator function of the $j$th feature being 'on', in other words, the corresponding ReLU being positive.

Let us introduce a shorthand notation $\hat{\theta}_{ij} = \theta_{z,ij} 1_{g_j>0}$. We can now write

$$BGS(x, y) = \sum_{i=1}^{C} (f(x;\theta)_i - y_i) \sum_j \hat{\theta}_{ij}. \tag{8}$$

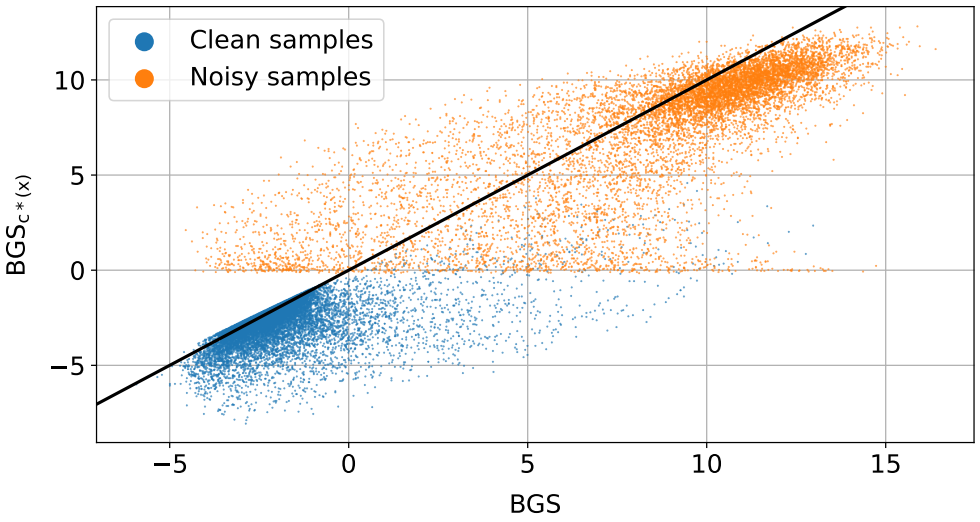


Figure 2: Scatter plot, where one dot is defined by the $BGS$ and $BGS_{c^*(x)}$ of a training sample, and the $BGS_{c^*(x)} = BGS$ line. The data belongs to the early stopping epoch during training a ResNet-18 network on CIFAR-10 with 20% uniform label noise. For the training setup see Section 6.

In addition, let $BGS_i(x, y)$ denote the $i$th member of the sum.

If all the features are active (the indicator function is always on) then the sign of the BGS does not depend on whether the example is noisy or not because in this case $\sum_j \hat{\theta}_{ij} = \sum_j \theta_{z,ij}$, which does not depend on the sample $(x, y)$ for any class $i$. In other words, the only dependence on $(x, y)$ is introduced by the fact that $x$ determines which features are active and which ones are turned off.

Let us now focus only on the logit of the true class $c^*(x)$ of an example $(x, y)$. We have $c(x) = c^*(x)$ if and only if the sample is clean. Now, $BGS_{c^*(x)}(x, y)$ *has an opposite sign for clean and noisy samples.* For a clean example, where $y_{c^*(x)} = 1$, the coefficient of the true class is $f(x; \theta)_{c^*(x)} - 1$, based on Equation (8), which is non-positive. Similarly, for an example with an incorrect label, where $y_{c^*(x)} = 0$, the coefficient that belongs to $c^*(x)$ is $f(x; \theta)_{c^*(x)}$, which is non-negative. This observation has also been made in Liu et al. (2023).

The remaining requirements for $BGS(x, y)$ to be positive if and only if the sample is noisy is to have $\sum_j \hat{\theta}_{c^*(x)j} > 0$ and $BGS_{c^*(x)}(x, y) \approx BGS(x, y)$. This is important, because $c^*(x)$ is unknown, but $BGS(x, y)$ can be easily computed. We find that both of these rather natural requirements are empirically supported, both implicitly (via the performance of the BGSS method on the label noise detection task) and explicitly. Figure 2 illustrates this, visualizing the relationship of $BGS(x, y)$ and $BGS_{c^*(x)}(x, y)$ in an example scenario. Indeed, the mass of the distribution is close to the $BGS(x, y) = BGS_{c^*(x)}(x, y)$ line and the decision based on the sign of BGS separates the clean examples reasonably well.

## 5. Training and Checkpoint Selection

Now that we have defined two methods for using a model checkpoint to filter noisy input samples (Prediction and BGSS), we need to decide on what checkpoints to select from the training. Here, our main insight is that the training algorithm, and in particular, the learning rate schedule has a large influence on the utility for early stopped checkpoints for noise identification.

**Training setup.** We experiment with the CIFAR-100 (Krizhevsky, 2009) dataset that contains 60,000 images that belong to 100 classes. The network architecture we used was ResNet-18 (He et al., 2016). Training was implemented with mini-batch SGD with a batch size of 128 for each dataset. Since the learning rate schedulers we are focusing on behave similarly on different learning tasks, we present only this setup here.

**Learning rate schedulers.** We experiment with three different learning rate schedules. The *multistep* scheduler uses an initial learning rate of 0.1 which is divided by 10 after epochs 100 and 150. The *linear* scheduler also sets the initial value of 0.1 that is linearly decreased to 0.001 during 200 epochs. The *restarted cosine* scheduler applies identical schedules for 5 consecutive 40 epoch intervals. In each interval a cosine scheduler decreases the learning rate from 0.1 to 0.001. That is, the learning rate in epoch $i \in \{1, \ldots, 200\}$ is $(1 - 0.001) \cdot \cos(\pi i/80) + 0.001$.

**Early stopping.** We will indicate the early stopping epoch given by the checkpoint with the best validation loss until the epoch when the validation loss has not improved for 20 consecutive epochs. To implement early stopping, we *do not need a clean dataset*. We use a held-out validation set from the same noisy distribution. In the case of the multistep scheduler, early stopping waited for at least the first learning rate drop. Note that we do not actually stop training here at the early stopping epoch, we always complete the 200 epochs for the sake of the analysis.

### 5.1. Empirical Observations

**Models from the memorization phase are useless.** The results are shown in Figure 3. We can observe that recall of noise detection dramatically decreases after the early stopping point for both detection methods. This is because memorization causes the model to classify more and more training examples correctly, eventually memorizing the entire training set, and the model becomes useless for predicting noise.

**The best generalizing model might be useless.** The double-descent phenomenon in time (Nakkiran et al., 2020) can be seen especially well in the case of the linear schedule. Here, the model checkpoint with the best generalization is useless for detecting noise. The restarted schedule acts very similarly to the linear schedule in each 40 epoch interval.

**The important checkpoints coincide only for multistep.** Most importantly, for continuous schedules like the linear or the cosine schedule, the different critical points: early stopping, best precision and best recall, might be far from each other. At the same time, in the case of a well-calibrated multi-step scheduler these important points all coincide.

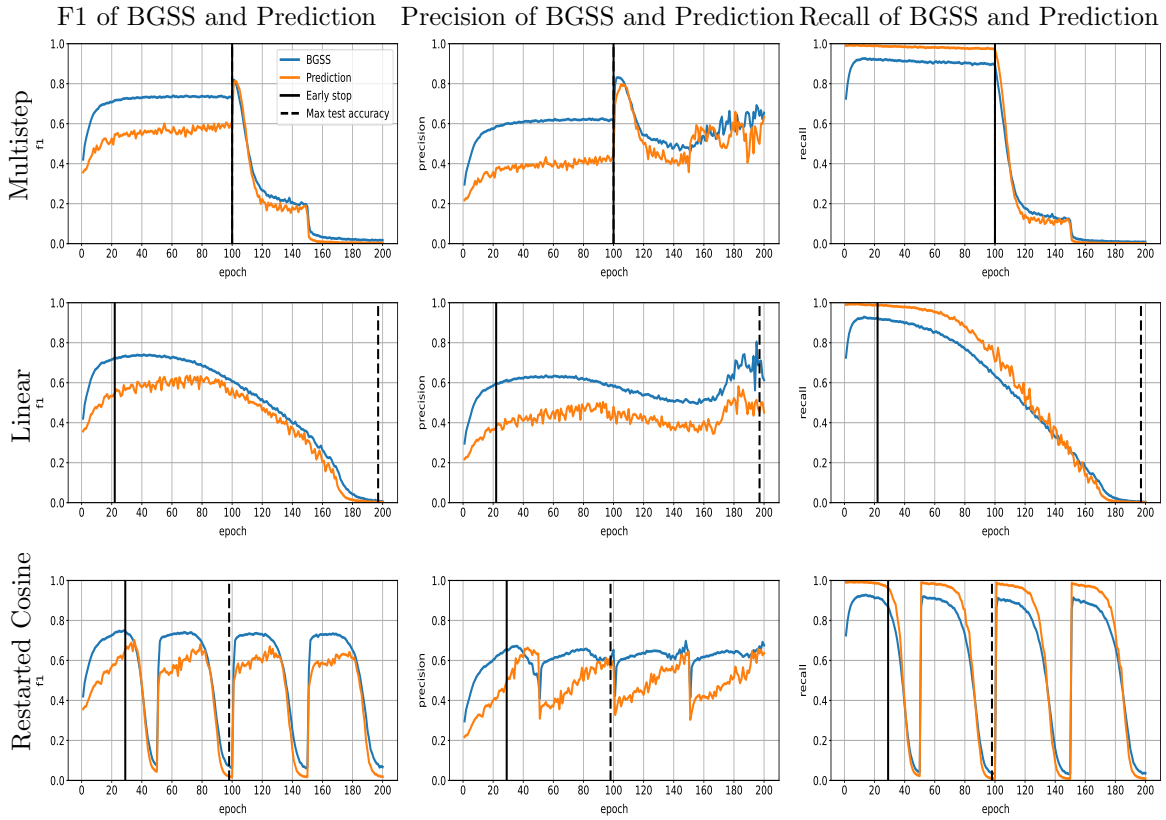


Figure 3: CIFAR-100, 20% uniform noise, with three schedulers: multistep, linear, and restarted cosine. Metrics of noise detection over the training data, when using the model checkpoint at the given training epoch, are shown as a function of training epochs for our two methods: Prediction and BGSS (see Section 4). Regarding the CIFAR-100 training dynamics, the early stopping epoch and the maximum test accuracy epoch over all the 200 epochs are also shown.

### 5.2. Design Decisions

**We need a multistep scheduler.** It is rather clear that our approach works best with a learning rate scheduler where the best checkpoints according to the different metrics (precision, recall, and F1 measure of noise detection, and test accuracy) coincide. This is the case if the multistep scheduler is used.

**We avoid the second descent.** It is important that we need the first local early stopping point, and not the best generalizing model. This is because the learning dynamics can exhibit the double descent phenomenon in time (Nakkiran et al., 2020), and in this case, the best generalization could be the result of the second descent. However, at that point, the training data has already been memorized and generalization is due to the so-called benign overfitting phenomenon (Sanyal et al., 2021; Bartlett et al., 2020).

**We aggregate over the best $k$ models.** Since the metrics are clearly rather noisy, smoothing might be necessary. We select the best $k$ models according to the validation loss from the epoch checkpoints until the 20th epoch after the early stopping epoch (where 20 is the early stopping tolerance parameter). We then compute the predictions for all these models and aggregate the result. The aggregation method can be either the mean of the raw predicted values (the BGS or the predicted class distribution) or a majority vote. This technique resembles temporal ensembling (Laine and Aila, 2017).

Table 1: Accuracy, F1 measure, precision and recall of several methods over different datasets and noise distributions. Voting and mean ensemble aggregation is denoted by V and M, respectively. The best result is highlighted in green and in boldface in each setup. The two runner-up results are highlighted in yellow.

| | noise (%) | 10% uniform | | | | 20% uniform | | | | 10% flip | | | | 20% flip | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Acc. | F1 | Prec. | Recall | Acc. | F1 | Prec. | Recall | Acc. | F1 | Prec. | Recall | Acc. | F1 | Prec. | Recall |
| CIFAR-10 | ES-BGSS (T1) | 0.961 | 0.828 | 0.733 | 0.952 | 0.957 | 0.897 | 0.861 | 0.936 | 0.951 | 0.796 | 0.684 | 0.953 | 0.951 | 0.886 | 0.833 | 0.947 |
| | ES-BGSS (T5-V) | 0.979 | 0.902 | 0.858 | 0.951 | 0.974 | 0.935 | 0.930 | 0.939 | 0.974 | 0.881 | 0.819 | 0.954 | 0.970 | 0.929 | 0.907 | 0.952 |
| | ES-BGSS (T5-M) | 0.972 | 0.873 | 0.797 | 0.965 | 0.972 | 0.930 | 0.911 | 0.950 | 0.958 | 0.821 | 0.716 | 0.963 | 0.960 | 0.907 | 0.862 | 0.957 |
| | ES-BGSS (T10-V) | **0.985** | **0.924** | **0.922** | 0.925 | **0.976** | **0.938** | **0.959** | 0.917 | 0.982 | 0.913 | 0.895 | 0.933 | 0.975 | 0.938 | 0.941 | 0.936 |
| | ES-BGSS (T10-M) | 0.977 | 0.892 | 0.835 | 0.958 | 0.974 | 0.936 | 0.929 | 0.943 | 0.964 | 0.841 | 0.751 | 0.955 | 0.963 | 0.913 | 0.876 | 0.953 |
| | ES-Prediction (T1) | 0.961 | 0.835 | 0.731 | **0.972** | 0.956 | 0.897 | 0.833 | **0.972** | 0.966 | 0.851 | 0.754 | **0.976** | 0.965 | 0.919 | 0.867 | **0.977** |
| | ES-Prediction (T5-V) | 0.975 | 0.884 | 0.814 | 0.966 | 0.968 | 0.923 | 0.885 | 0.964 | 0.979 | 0.901 | 0.838 | 0.975 | 0.975 | 0.941 | 0.909 | 0.976 |
| | ES-Prediction (T5-M) | 0.977 | 0.892 | 0.828 | 0.966 | 0.969 | 0.925 | 0.891 | 0.962 | 0.980 | 0.907 | 0.849 | 0.973 | 0.976 | 0.944 | 0.914 | 0.975 |
| | ES-Prediction (T10-V) | 0.983 | 0.915 | 0.887 | 0.946 | 0.973 | 0.934 | 0.923 | 0.945 | 0.986 | 0.931 | 0.901 | 0.963 | **0.981** | **0.954** | **0.943** | 0.966 |
| | ES-Prediction (T10-M) | 0.983 | 0.918 | 0.890 | 0.947 | 0.974 | 0.935 | 0.926 | 0.945 | **0.987** | **0.935** | **0.906** | 0.965 | **0.981** | **0.954** | **0.943** | 0.965 |
| CIFAR-100 | ES-BGSS (T1) | 0.922 | 0.695 | 0.576 | 0.875 | 0.866 | 0.732 | 0.616 | 0.904 | 0.907 | 0.648 | 0.521 | 0.857 | 0.887 | 0.738 | 0.693 | 0.789 |
| | ES-BGSS (T5-V) | **0.951** | **0.771** | 0.736 | 0.809 | 0.921 | 0.828 | 0.738 | 0.943 | 0.945 | 0.753 | 0.686 | 0.833 | **0.917** | **0.788** | 0.809 | 0.769 |
| | ES-BGSS (T5-M) | 0.941 | 0.744 | 0.665 | 0.844 | 0.906 | 0.805 | 0.695 | 0.957 | 0.927 | 0.702 | 0.593 | 0.860 | 0.906 | 0.773 | 0.753 | 0.794 |
| | ES-BGSS (T10-V) | **0.951** | 0.725 | **0.833** | 0.642 | **0.949** | **0.872** | **0.893** | 0.852 | **0.955** | **0.769** | **0.796** | 0.743 | 0.913 | 0.759 | **0.865** | 0.676 |
| | ES-BGSS (T10-M) | 0.942 | 0.731 | 0.688 | 0.779 | 0.939 | 0.860 | 0.804 | 0.924 | 0.932 | 0.708 | 0.621 | 0.824 | 0.906 | 0.765 | 0.767 | 0.764 |
| | ES-Prediction (T1) | 0.903 | 0.665 | 0.513 | **0.943** | 0.735 | 0.600 | 0.432 | 0.982 | 0.904 | 0.653 | 0.509 | **0.908** | 0.889 | 0.756 | 0.677 | **0.855** |
| | ES-Prediction (T5-V) | 0.934 | 0.732 | 0.622 | 0.888 | 0.770 | 0.636 | 0.468 | **0.993** | 0.932 | 0.710 | 0.616 | 0.837 | 0.901 | 0.758 | 0.745 | 0.772 |
| | ES-Prediction (T5-M) | 0.939 | 0.744 | 0.650 | 0.872 | 0.853 | 0.729 | 0.581 | 0.980 | 0.936 | 0.722 | 0.640 | 0.828 | 0.905 | 0.763 | 0.764 | 0.762 |
| | ES-Prediction (T10-V) | 0.945 | 0.727 | 0.734 | 0.720 | 0.919 | 0.824 | 0.733 | 0.941 | 0.944 | 0.713 | 0.734 | 0.693 | 0.896 | 0.706 | 0.823 | 0.618 |
| | ES-Prediction (T10-M) | 0.945 | 0.717 | 0.757 | 0.682 | 0.929 | 0.837 | 0.781 | 0.903 | 0.948 | 0.725 | 0.759 | 0.694 | 0.898 | 0.707 | 0.835 | 0.612 |
| Tiny ImageNet | ES-BGSS (T1) | 0.791 | 0.472 | 0.317 | 0.921 | 0.833 | 0.682 | 0.550 | 0.898 | 0.745 | 0.393 | 0.259 | 0.815 | 0.736 | 0.524 | 0.410 | 0.724 |
| | ES-BGSS (T5-V) | 0.839 | 0.543 | 0.381 | 0.944 | 0.878 | 0.752 | 0.634 | 0.923 | 0.799 | 0.462 | 0.317 | 0.851 | 0.786 | 0.588 | 0.479 | 0.760 |
| | ES-BGSS (T5-M) | 0.804 | 0.497 | 0.336 | 0.953 | 0.865 | 0.734 | 0.605 | 0.933 | 0.739 | 0.403 | 0.263 | 0.870 | 0.749 | 0.554 | 0.430 | 0.779 |
| | ES-BGSS (T10-V) | **0.873** | **0.599** | **0.440** | 0.935 | **0.902** | **0.789** | **0.695** | 0.912 | **0.838** | **0.506** | **0.367** | 0.818 | **0.819** | **0.617** | **0.536** | 0.726 |
| | ES-BGSS (T10-M) | 0.810 | 0.504 | 0.343 | 0.952 | 0.874 | 0.747 | 0.622 | 0.934 | 0.739 | 0.404 | 0.263 | 0.873 | 0.755 | 0.563 | 0.438 | 0.787 |
| | ES-Prediction (T1) | 0.646 | 0.363 | 0.222 | 0.994 | 0.653 | 0.534 | 0.365 | **0.995** | 0.674 | 0.377 | 0.234 | 0.975 | 0.686 | 0.552 | 0.387 | 0.966 |
| | ES-Prediction (T5-V) | 0.668 | 0.378 | 0.234 | **0.995** | 0.667 | 0.545 | 0.375 | **0.995** | 0.699 | 0.398 | 0.250 | **0.983** | 0.706 | 0.569 | 0.403 | **0.968** |
| | ES-Prediction (T5-M) | 0.695 | 0.398 | 0.249 | 0.994 | 0.689 | 0.560 | 0.390 | 0.994 | 0.732 | 0.425 | 0.272 | 0.978 | 0.732 | 0.590 | 0.426 | 0.962 |
| | ES-Prediction (T10-V) | 0.706 | 0.407 | 0.256 | **0.995** | 0.705 | 0.574 | 0.404 | 0.994 | 0.741 | 0.434 | 0.279 | 0.977 | 0.742 | 0.600 | 0.436 | 0.962 |
| | ES-Prediction (T10-M) | 0.735 | 0.432 | 0.276 | 0.992 | 0.739 | 0.603 | 0.433 | 0.992 | 0.775 | 0.466 | 0.307 | 0.971 | 0.774 | **0.629** | 0.469 | 0.952 |

Table 2: Accuracy, F1 measure, precision and recall of several methods over different datasets and noise distributions. ES-BGSS and ES-Prediction are our proposed methods. The best result is highlighted in green and in boldface in each setup. The two runner-up results are highlighted in yellow.

| | noise (%) | 10% uniform | | | | 20% uniform | | | | 10% flip | | | | 20% flip | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Acc. | F1 | Prec. | Recall | Acc. | F1 | Prec. | Recall | Acc. | F1 | Prec. | Recall | Acc. | F1 | Prec. | Recall |
| CIFAR-10 | ES-BGSS (T10-V) | **0.985** | **0.924** | **0.922** | 0.925 | **0.976** | **0.938** | **0.959** | 0.917 | 0.982 | 0.913 | 0.895 | 0.933 | 0.975 | 0.938 | 0.941 | 0.936 |
| | ES-Prediction (T10-M) | 0.983 | 0.918 | 0.890 | 0.947 | 0.974 | 0.935 | 0.926 | 0.945 | **0.987** | **0.935** | **0.906** | 0.965 | **0.981** | **0.954** | **0.943** | 0.965 |
| | AUM | 0.971 | 0.870 | 0.791 | 0.966 | 0.966 | 0.919 | 0.882 | 0.960 | 0.971 | 0.869 | 0.798 | 0.954 | 0.958 | 0.896 | 0.888 | 0.905 |
| | CL ($C_{\tilde{y},y^*}$) | 0.960 | 0.806 | 0.780 | 0.835 | 0.940 | 0.849 | 0.850 | 0.848 | 0.960 | 0.815 | 0.761 | 0.877 | 0.949 | 0.876 | 0.860 | 0.892 |
| | CL (C+NR) | 0.967 | 0.846 | 0.791 | 0.910 | 0.952 | 0.886 | 0.848 | 0.927 | 0.958 | 0.806 | 0.747 | 0.875 | 0.943 | 0.863 | 0.845 | 0.881 |
| | SimiFeat-V | 0.957 | 0.821 | 0.701 | **0.990** | 0.962 | 0.912 | 0.847 | 0.989 | 0.956 | 0.818 | 0.696 | 0.992 | 0.958 | 0.905 | 0.839 | 0.982 |
| | SimiFeat-R | 0.958 | 0.826 | 0.709 | **0.990** | 0.960 | 0.909 | 0.839 | **0.991** | 0.956 | 0.818 | 0.694 | **0.994** | 0.950 | 0.889 | 0.812 | **0.983** |
| CIFAR-100 | ES-BGSS (T10-V) | **0.951** | 0.725 | **0.833** | 0.642 | 0.949 | 0.872 | **0.893** | 0.852 | **0.955** | **0.769** | **0.796** | 0.743 | **0.913** | **0.759** | **0.865** | 0.676 |
| | ES-Prediction (T10-M) | 0.945 | 0.717 | 0.757 | 0.682 | 0.929 | 0.837 | 0.781 | 0.903 | 0.948 | 0.725 | 0.759 | 0.694 | 0.898 | 0.707 | 0.835 | 0.612 |
| | AUM | 0.945 | **0.783** | 0.654 | 0.974 | **0.951** | **0.888** | 0.821 | 0.965 | 0.929 | 0.721 | 0.593 | 0.919 | 0.871 | 0.677 | 0.684 | 0.671 |
| | CL ($C_{\tilde{y},y^*}$) | 0.833 | 0.474 | 0.349 | 0.738 | 0.791 | 0.581 | 0.490 | 0.714 | 0.839 | 0.476 | 0.353 | 0.732 | 0.797 | 0.594 | 0.497 | 0.737 |
| | CL (C+NR) | 0.844 | 0.518 | 0.377 | 0.828 | 0.817 | 0.648 | 0.530 | 0.832 | 0.847 | 0.507 | 0.374 | 0.789 | 0.797 | 0.592 | 0.496 | 0.732 |
| | SimiFeat-V | 0.847 | 0.568 | 0.398 | 0.989 | 0.870 | 0.754 | 0.611 | 0.986 | 0.844 | 0.556 | 0.388 | 0.983 | 0.854 | 0.723 | 0.586 | 0.944 |
| | SimiFeat-R | 0.764 | 0.462 | 0.300 | **0.997** | 0.790 | 0.658 | 0.491 | **0.998** | 0.764 | 0.456 | 0.296 | **0.994** | 0.769 | 0.629 | 0.464 | **0.976** |
| Tiny ImageNet | ES-BGSS (T10-V) | **0.873** | **0.599** | **0.440** | 0.935 | **0.902** | **0.789** | **0.695** | 0.912 | 0.838 | 0.506 | 0.367 | 0.818 | 0.819 | 0.617 | 0.536 | 0.726 |
| | ES-Prediction (T10-M) | 0.735 | 0.432 | 0.276 | 0.992 | 0.739 | 0.603 | 0.433 | 0.992 | 0.775 | 0.466 | 0.307 | 0.971 | 0.774 | 0.629 | 0.469 | 0.952 |
| | AUM | 0.847 | 0.561 | 0.395 | 0.966 | 0.852 | 0.722 | 0.579 | 0.959 | 0.837 | **0.531** | 0.375 | 0.913 | 0.790 | 0.576 | 0.484 | 0.711 |
| | CL ($C_{\tilde{y},y^*}$) | 0.734 | 0.332 | 0.223 | 0.652 | 0.688 | 0.464 | 0.353 | 0.674 | 0.736 | 0.340 | 0.228 | 0.671 | 0.700 | 0.468 | 0.363 | 0.660 |
| | CL (C+NR) | 0.751 | 0.389 | 0.259 | 0.784 | 0.732 | 0.538 | 0.411 | 0.781 | 0.746 | 0.374 | 0.249 | 0.749 | 0.700 | 0.466 | 0.363 | 0.653 |
| | SimiFeat-V | 0.847 | 0.541 | 0.389 | 0.886 | 0.869 | 0.716 | 0.632 | 0.826 | **0.842** | 0.527 | **0.378** | 0.867 | **0.843** | **0.662** | **0.584** | 0.764 |
| | SimiFeat-R | 0.554 | 0.312 | 0.185 | **0.999** | 0.596 | 0.497 | 0.331 | **0.998** | 0.551 | 0.310 | 0.184 | **0.996** | 0.577 | 0.482 | 0.320 | **0.982** |

## 6. Empirical Evaluation of Noise Detection

Armed with the design choices presented so far, we explore the hyperparameters of ensemble smoothing, and perform an exstensive comparison of our approach to several competing solutions for noise detection from related work over different noise models and datasets.

### 6.1. Experimental Setup

The setup was the same as in Section 3, but here we used only the multistep scheduler. We experiment with three datasets: CIFAR-10 and CIFAR-100 (Krizhevsky, 2009), and Tiny ImageNet (Le and Yang, 2015). The CIFAR datasets contain 60,000 images that belong to 10 and 100 classes, respectively. The Tiny ImageNet dataset contains 100,000 images divided into 200 classes.

The presented metrics are based on a single run in each setup. Our complete empirical evaluation took approximately 1000 GPU hours. We used mainly NVIDIA RTX 3070 and A6000 GPUs. For efficiency, the BGS values were computed during the underlying training. That is, we used live gradients on training batches from the given epoch and not saved checkpoints.

**Noise models.** On all the datasets, we implemented the two most common noise models applied in related work. These were the *uniform* model and the *flip* model (also called symmetric and asymmetric, respectively). In the uniform model $p_{noisy}(y|x)$ is selected to be the uniform distribution over the labels other than the ground truth label in the original clean dataset. In the flip model the labels of the noisy samples are shifted by one class relative to the ground truth labels. That is, if the ground truth class is $c$ then the noisy class will be $c + 1 \mod C$, where $C$ is the number of classes.

### 6.2. Ensemble Smoothing

So far, we have focused only on predictions based on individual models. Here, we explore the benefit of using more than one model in an ensemble fashion.

Given an ensemble size $k > 0$, we define the members of the ensemble to be the top-$k$ models based on validation loss, selected from the epoch checkpoints up until 20 epochs after the early stopping point (that is, from all the available epochs after the early stopped training).

There are two hyperparameters: the number of epochs $k$ used for smoothing, and the aggregation method. Table 1 compares the performance of the combination of three values of $k$ for the number of top-$k$ epochs: 1, 5, and 10, and two aggregation methods: mean and voting.

The detection methods combined with the model selection method described above define a noise detection solution. From now on, we shall call these solutions ES-Prediction and ES-BGSS, where ES stands for early stopping.

**Top-10 is a good choice.** In general, aggregating a larger number of epochs seems to offer a better performance. However, this number has a limit, because we need high quality models from around the early stopping point. Indeed, for CIFAR-100 using only the top-5 models is sometimes preferable. In this sense, using the top-10 models is a robust choice.

**The aggregation method.** For ES-BGSS the voting method is clearly preferable, while for ES-Prediction the mean is consistently better. This could be due to the fact that ES-BGSS works with gradients that could have more extreme outliers than the softmax vector has that is used by ES-Prediction.

**Comparing ES-BGSS and ES-Prediction.** ES-Prediction is better in terms of recall overall, and tends to be weaker in precision. This is consistent with Figure 3. This indicates that most examples with noisy labels are misclassified, but there are some clean examples that are misclassified as well. ES-BGSS is better at differentiating between hard clean examples and noisy examples, which also shows in the better aggregated metrics (accuracy and F1 measure), with some exceptions in the case of flip noise.

### 6.3. Comparison with Related Work

Let us now compare our method with a few well-known recent baselines from related work. For all the methods we include, we used the implementations provided by the authors, along with the proposed hyperparameters for the datasets in our evaluation. In the following, we briefly describe the algorithms we used.

**Area under Margin (AUM) (Pleiss et al., 2020).** The method is based on integrating the logit margin over the training epochs. In addition, extra training samples are added in order to estimate the correct decision threshold.

**Confident Learning (CL) (Northcutt et al., 2021).** Here, the main idea is to statistically approximate the so called confident joint $C_{\tilde{y},y^*}$ that determines the distribution of the label errors in detail, using a prior and the prediction errors of a model trained on noisy samples. Several statistical techniques are applied to improve the results. The method assumes that the tested sample is not in the training set so several rounds of training are necessary on different splits so that for each sample we have at least one independent model. Apart from the method based on $C_{\tilde{y},y^*}$, we also include a variant called $C + NR$ that also performed well.

**SimiFeat (Zhu et al., 2022).** This approach relies purely on clustering using a pre-trained foundation model that computes features for the input examples. We include both the voting and ranking-based variants in our comparison. The features were provided by the ViT-B/32 (Dosovitskiy et al., 2021) feature extractor used by the authors, pre-trained by CLIP (Radford et al., 2021). We stress here, that all the other methods, including ours, use *only the noisy dataset* and no additional external knowledge.

**Discussion.** Table 2 contains our experimental results. Clearly, ES-BGSS is competitive with the rest of the methods, in many cases winning important categories like the aggregated F1 score. It is quite surprising that ES-Prediction is also rather competitive, beating specialized methods in several cases. SimiFeat is clearly the best in terms of recall, but at the cost of a relatively low precision. In fact, considering Table 1, with smaller ensembles the ES-Prediction method is competitive with SimiFeat in terms of recall as well. Clearly, the method of choice depends on the requirements of the specific applications at hand, where precision and recall might have different priorities.

## 7. Empirical Evaluation of Noise-Filtered Training

We also evaluate our noise detection approach in the context of noise-filtered training. Note that robust training under label noise is a specialized area with a large literature, where the methods often involve a modified training algorithm, adaptive filtering or modification of samples, and so on. A few examples include layer-wise early topping (Bai et al., 2021), outlier detection during training based on the gradients (Yang et al., 2022; Sedova et al., 2023), logit adjustment during training (Fu et al., 2023), assessing the influence of examples on the training (Kwon and Zou, 2022; Pruthi et al., 2020) and so on. A recent survey can be found in (Song et al., 2023).

Here, we focused specifically on noise detection, but it is still interesting to evaluate the performance of the different methods when applied on their own as a noise filter before normal vanilla training. That is, we apply each noise detector on the training set, remove the examples that are flagged noisy and run the training algorithm on the cleansed training set.

Table 3: Filtered test accuracy (F.A.), and noise detection metrics: accuracy, F1 measure, precision and recall of several methods over CIFAR-10N and CIFAR-100N. The best result is highlighted in green and in boldface in each setup. The runner-up result is highlighted in yellow.

| | CIFAR-10N | | | | | CIFAR-100N | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | F.A. | Acc. | F1 | Prec. | Rec. | F.A. | Acc. | F1 | Prec. | Rec. |
| Clean | 0.95 | - | - | - | - | 0.77 | - | - | - | - |
| Noisy | 0.91 | 0.00 | 0.00 | 0.00 | 0.00 | 0.62 | 0.00 | 0.00 | 0.00 | 0.00 |
| Ground Truth | 0.94 | 1.00 | 1.00 | 1.00 | 1.00 | 0.71 | 1.00 | 1.00 | 1.00 | 1.00 |
| ES-Pred. (T1) | 0.91 | 0.93 | 0.78 | 0.86 | 0.72 | **0.64** | **0.82** | 0.76 | 0.81 | 0.71 |
| ES-Pred. (T5-V) | **0.93** | 0.95 | 0.86 | 0.87 | 0.85 | 0.63 | 0.80 | 0.72 | 0.84 | 0.63 |
| ES-Pred. (T5-M) | 0.92 | 0.95 | 0.86 | 0.87 | 0.84 | 0.63 | 0.80 | 0.71 | 0.85 | 0.61 |
| ES-Pred. (T10-V) | 0.92 | 0.95 | 0.85 | 0.91 | 0.80 | 0.62 | 0.77 | 0.63 | 0.87 | 0.49 |
| ES-Pred. (T10-M) | 0.92 | 0.95 | 0.85 | 0.91 | 0.79 | 0.61 | 0.75 | 0.59 | 0.88 | 0.44 |
| ES-BGSS (T1) | 0.91 | 0.92 | 0.74 | 0.89 | 0.63 | 0.60 | 0.73 | 0.58 | 0.80 | 0.45 |
| ES-BGSS (T5-V) | 0.91 | 0.93 | 0.78 | **0.94** | 0.67 | 0.59 | 0.72 | 0.53 | 0.79 | 0.40 |
| ES-BGSS (T5-M) | 0.91 | 0.94 | 0.80 | 0.92 | 0.70 | 0.59 | 0.71 | 0.53 | 0.77 | 0.41 |
| ES-BGSS (T10-V) | 0.91 | 0.94 | 0.81 | **0.94** | 0.71 | 0.58 | 0.69 | 0.44 | 0.78 | 0.31 |
| ES-BGSS (T10-M) | 0.92 | 0.94 | 0.83 | 0.92 | 0.75 | 0.58 | 0.69 | 0.49 | 0.73 | 0.37 |
| CL ($C_{\tilde{y},y^*}$) | 0.91 | 0.89 | 0.72 | 0.66 | 0.79 | 0.60 | 0.73 | 0.67 | 0.67 | 0.67 |
| CL (C+NR) | 0.91 | 0.90 | 0.75 | 0.67 | 0.85 | 0.60 | 0.76 | 0.69 | 0.71 | 0.67 |
| AUM | 0.91 | 0.87 | 0.69 | 0.60 | 0.83 | 0.63 | 0.78 | 0.65 | **0.89** | 0.52 |
| SimiFeat-V | **0.93** | **0.96** | **0.89** | 0.86 | **0.93** | **0.64** | **0.82** | 0.77 | 0.81 | 0.74 |
| SimiFeat-R | **0.93** | **0.96** | **0.89** | 0.87 | 0.91 | **0.64** | **0.82** | **0.80** | 0.74 | **0.86** |

### 7.1. Experimental Setup

The datasets we used were two recent human-annotated benchmarks proposed by Wei et al. (Wei et al., 2022): CIFAR-10N and CIFAR-100N. These contain labels given by humans, and a label is considered noisy if it is different from the official CIFAR10 and CIFAR100 label. More precisely, we used the 'fine' variant of CIFAR-100N that contains 40.2% noise and the 'random2' variant of CIFAR-10N that contains 18.12% noise. The training setup was otherwise identical to the one used in Section 6. Each configuration is measured in a single run.

### 7.2. Results

Table 3 contains the results. The 'Clean' row is the original clean CIFAR10/100 result trained without noisy labels. The rest of the table uses the CIFAR-10N/100N labels from (Wei et al., 2022). The 'Noisy' row represents the extreme case when we detect no noisy samples, so no samples are removed from the training set. The 'Ground Truth' method represents the case of perfect detection when exactly the noisy samples are removed from the training set.

**ES-Prediction is competitive.** It is clear from the table that in terms of test accuracy on the CIFAR tasks after filtering (column 'A.F.'), the ES-Prediction method is capable of

reaching the same performance as SimiFeat, despite of the fact that SimiFeat has access to a powerful foundation model, while our method uses only information from the training data. Again, the key to achieve this result lies in the selection of model checkpoints, based on early stopping. On CIFAR-100N, ES-BGSS is not among the best methods, indicating that ES-Prediction is more robust to this type of noise.

**Early stopping can beat filtering.** It is also striking that *filtering can actually decrease the model quality* compared to the case when all the samples are included in the training data. Note that when there is no explicit filtering, the noise is still filtered implicitly due to the application of early stopping in our training setup (Liu et al., 2023; Li et al., 2020).

## 8. Conclusions and Limitations

We have demonstrated that merely observing normal training and using a carefully selected set of model checkpoints using early stopping can lead to competitive noisy label detection across several settings. Even the simple and natural method of using the predictions of the selected checkpoints to filter noisy samples is very competitive in both synthetic and natural noise scenarios.

This is somewhat surprising, because state-of-the-art methods use specialized techniques such as optimizing thresholds, external foundation models, or dedicated training procedures, yet these extra techniques do not seem to yield a convincing advantage.

The key idea behind achieving these results is to carefully select an appropriate early stopping checkpoint and to apply smoothing over the noisy prediction by using several good checkpoints around the early stopping point.

As for limitations, our methods rely on the underlying training procedure. We have demonstrated that certain components such as the learning rate schedule have a very significant influence on the quality of the early stopping point in terms of noise filtering.

While the multistep scheduler worked very well in the scenarios we studied, in order to understand the scope of our method better, we could benefit from further theoretical progress on understanding implicit noise filtering beyond the present state-of-the-art, because currently the influence of learning rate schedulers is not yet taken into account (Liu et al., 2023; Li et al., 2020).

## Acknowledgments

This work was supported by the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory and project TKP2021-NVA-09, implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021-NVA funding scheme.

## References

Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron C. Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In Doina

Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 233–242. PMLR, 2017. URL http://proceedings.mlr.press/v70/arpit17a.html.

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. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 24392–24403, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/cc7e2b878868cbae992d1fb743995d8f-Abstract.html.

Peter L. Bartlett, Philip M. Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020. doi: 10.1073/pnas.1907378117. URL https://www.pnas.org/doi/abs/10.1073/pnas.1907378117.

Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. Understanding and Utilizing Deep Neural Networks Trained with Noisy Labels. In *Proceedings of the 36th International Conference on Machine Learning*, pages 1062–1070. PMLR, May 2019. URL https://proceedings.mlr.press/v97/chen19g.html. ISSN: 2640-3498.

Hao Cheng, Zhaowei Zhu, Xingyu Li, Yifei Gong, Xing Sun, and Yang Liu. Learning with instance-dependent label noise: A sample sieve approach. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=2VXyy9mIyU3.

Yinpeng Dong, Ke Xu, Xiao Yang, Tianyu Pang, Zhijie Deng, Hang Su, and Jun Zhu. Exploring memorization in adversarial training. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=7gE9V9GBZaI.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Boyi Fu, Yuncong Peng, and Xiaolin Qin. Learning with noisy labels via logit adjustment based on gradient prior method. *Applied Intelligence*, 53(20):24393–24406, October 2023. ISSN 1573-7497. doi: 10.1007/s10489-023-04609-1. URL https://doi.org/10.1007/s10489-023-04609-1.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems*, volume 31. Curran

Associates, Inc., 2018. URL https://papers.nips.cc/paper_files/paper/2018/hash/a19744e268754fb0148b017647355b7b-Abstract.html.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, volume 9908 of *Lecture Notes in Computer Science*, pages 630–645. Springer, 2016. doi: 10.1007/978-3-319-46493-0_38. URL https://doi.org/10.1007/978-3-319-46493-0_38.

Jinchi Huang, Lie Qu, Rongfei Jia, and Binqiang Zhao. O2U-Net: A Simple Noisy Label Detection Approach for Deep Neural Networks. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3325–3333, October 2019. doi: 10.1109/ICCV.2019.00342. URL https://ieeexplore.ieee.org/document/9008796. ISSN: 2380-7504.

Qingrui Jia, Xuhong Li, Lei Yu, Jiang Bian, Penghao Zhao, Shupeng Li, Haoyi Xiong, and Dejing Dou. Learning from training dynamics: Identifying mislabeled data beyond manually designed features. In Brian Williams, Yiling Chen, and Jennifer Neville, editors, *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 8041–8049. AAAI Press, 2023. doi: 10.1609/AAAI.V37I7.25972. URL https://doi.org/10.1609/aaai.v37i7.25972.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. MentorNet: Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2304–2313. PMLR, July 2018. URL https://proceedings.mlr.press/v80/jiang18c.html. ISSN: 2640-3498.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Yongchan Kwon and James Zou. Beta Shapley: a Unified and Noise-reduced Data Valuation Framework for Machine Learning. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pages 8780–8802. PMLR, May 2022. URL https://proceedings.mlr.press/v151/kwon22a.html. ISSN: 2640-3498.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=BJ6oOfqge.

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge. Technical report, Stanford University, 2015. URL http://vision.stanford.edu/teaching/cs231n/reports/2015/pdfs/yle_project.pdf. CS 231N report.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In Silvia Chiappa and Roberto Calandra, editors, *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 4313–4324. PMLR, 2020. URL http://proceedings.mlr.press/v108/li20j.html.

Chaoyue Liu, Amirhesam Abedsoltan, and Mikhail Belkin. On emergence of clean-priority learning in early stopped neural networks. *CoRR*, abs/2306.02533, 2023. doi: 10.48550/arXiv.2306.02533. URL https://doi.org/10.48550/arXiv.2306.02533.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=B1g5sA4twr.

Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. Confident learning: Estimating uncertainty in dataset labels. *J. Artif. Intell. Res.*, 70:1373–1411, 2021. doi: 10.1613/jair.1.12125. URL https://doi.org/10.1613/jair.1.12125.

Geoff Pleiss, Tianyi Zhang, Ethan R. Elenberg, and Kilian Q. Weinberger. Identifying mislabeled data using the area under the margin ranking. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/c6102b3727b2a7d8b1bb6981147081ef-Abstract.html.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/e6385d39ec9394f2f3a354d9d2b88eec-Abstract.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 2021. URL http://proceedings.mlr.press/v139/radford21a.html.

Amartya Sanyal, Puneet K. Dokania, Varun Kanade, and Philip H. S. Torr. How benign is benign overfitting ? In *9th International Conference on Learning Representations,*

*ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=g-wu9TMPODo.

Anastasiia Sedova, Lena Zellinger, and Benjamin Roth. Learning with noisy labels by adaptive gradient-based outlier removal. In Danai Koutra, Claudia Plant, Manuel Gomez-Rodriguez, Elena Baralis, and Francesco Bonchi, editors, *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part I*, volume 14169 of *Lecture Notes in Computer Science*, pages 237–253. Springer, 2023. doi: 10.1007/978-3-031-43412-9_14. URL https://doi.org/10.1007/978-3-031-43412-9_14.

Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE Trans. Neural Networks Learn. Syst.*, 34(11):8135–8153, 2023. doi: 10.1109/TNNLS.2022.3152527. URL https://doi.org/10.1109/TNNLS.2022.3152527.

Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. Learning with noisy labels revisited: A study using real-world human annotations. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=TBWA6PLJZQm.

Xingyi Yang, Daquan Zhou, Songhua Liu, Jingwen Ye, and Xinchao Wang. Deep model reassembly. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 25739–25753. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/a4e683f0ce6b91e7fbdae9d32642d88f-Paper-Conference.pdf.

Chenglin Yu, Xinsong Ma, and Weiwei Liu. Delving into Noisy Label Detection with Clean Data. In *Proceedings of the 40th International Conference on Machine Learning*, pages 40290–40305. PMLR, July 2023. URL https://proceedings.mlr.press/v202/yu23b.html. ISSN: 2640-3498.

Zhaowei Zhu, Zihao Dong, and Yang Liu. Detecting corrupted labels without training a model to predict. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 27412–27427. PMLR, June 2022. URL https://proceedings.mlr.press/v162/zhu22a.html. ISSN: 2640-3498.