

---

# Neural Relation Graph: A Unified Framework for Identifying Label Noise and Outlier Data

---

**Jang-Hyun Kim**  
Seoul National University  
janghyun@mllab.snu.ac.kr

**Sangdoo Yun**  
NAVER AI Lab  
sangdoo.yun@navercorp.com

**Hyun Oh Song**  
Seoul National University  
hyunoh@mllab.snu.ac.kr

## Abstract

Diagnosing and cleaning data is a crucial step for building robust machine learning systems. However, identifying problems within large-scale datasets with real-world distributions is challenging due to the presence of complex issues such as label errors, under-representation, and outliers. In this paper, we propose a unified approach for identifying the problematic data by utilizing a largely ignored source of information: a relational structure of data in the feature-embedded space. To this end, we present scalable and effective algorithms for detecting label errors and outlier data based on the relational graph structure of data. We further introduce a visualization tool that provides contextual information of a data point in the feature-embedded space, serving as an effective tool for interactively diagnosing data. We evaluate the label error and outlier/out-of-distribution (OOD) detection performances of our approach on the large-scale image, speech, and language domain tasks, including ImageNet, ESC-50, and MNLI. Our approach achieves state-of-the-art detection performance on all tasks considered and demonstrates its effectiveness in debugging large-scale real-world datasets across various domains.

## 1 Introduction

Identifying problems within datasets is crucial for improving the robustness of machine learning systems and analyzing the model failures [43]. For instance, identifying mislabeled or uninformative data help construct concise and effective training datasets [31], while identifying whether test data is OOD or corrupted allows for more accurate model evaluation and analysis [52].

In recent years, efforts have been made to identify problematic data by utilizing unary scores on individual data from trained models, such as estimating data influence [21], monitoring prediction variability throughout training [50], and calculating prediction error margins [32]. However, identifying such data can be challenging, particularly when dealing with large-scale datasets from real-world distributions. In real-world settings, datasets may have complex problems, including label issues, under-representation, and outliers, each of which can lead to the model error and prediction sensitivity [22]. For example, Figure 2 shows that a neural network exhibits low negative prediction margins and high loss values for both a sample with label error and outlier data, indicating that previous unary scoring methods may have limitations in identifying the sources of problems.

In this work, we propose a unified framework for identifying label errors and outliers by leveraging the feature-embedded structure of a dataset that provides richer information than individual data alone [45, 34]. We measure the relationship among data in the feature embedding space while comparing the assigned labels independently. By comparing input data and labels separately, we are able to isolate the factors contributing to model errors, resulting in improved detection of the problematic data. Based on this relational information, we construct a novel graph structure on the dataset and identify whether the data itself or the label is problematic (Figure 1). To this end, we develop scalable graph algorithms that efficiently identify label errors and outlier data points.

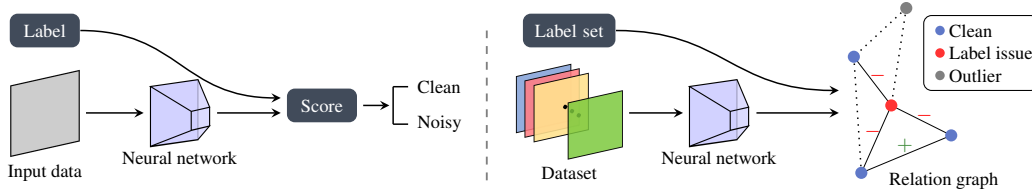


Figure 1: The conceptual illustration of the conventional approaches (top) and our proposed approach (bottom). While the previous approaches measure the prediction error or sensitivity on each data point, our method identifies problematic data by leveraging the relational structure of data. In the relation graph, positive edges signify complementary relation, negative edges denote conflicting relation, and dashed lines indicate negligible relation between data.

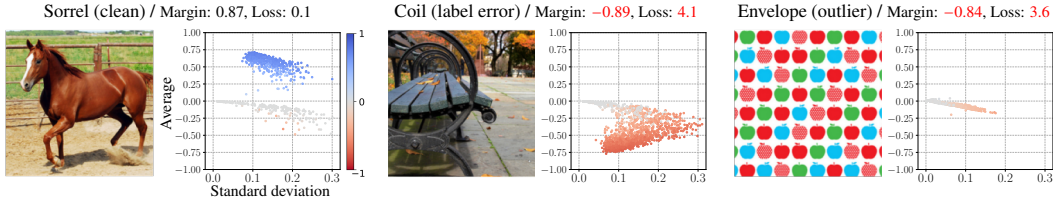


Figure 2: Data samples and their labels from ImageNet and the corresponding relation maps by an MAE-Large model [13]. We report the prediction margin score ( $\in [-1, 1]$ ) and the loss value next to the label. The color represents the relation value at the last converged checkpoint. We present the detailed procedure for generating the relation maps in Section 3.4.

In Section 3.4, we further introduce a tool named data relation map that provides contextual information on data points within the feature-embedded structure. This tool visualizes the relational structure of a data sample and can serve as an interactive tool for diagnosing data. Specifically, a data relation map measures the variance and mean value of relations between a data pair throughout training. From Figure 2, we observe that the second and the third samples exhibit different relation map patterns, although they have similar margin and loss scores. This highlights that the relational structure provides complementary information not captured by the unary scoring methods.

Our approach only requires the model’s feature embedding and prediction score on data, making it more scalable compared to methods that require calculating the network gradient on each data point or retraining models multiple times to estimate data influence [37, 17]. Furthermore, our method is domain- and model-agnostic, and thus is applicable to various tasks. We evaluate our approach on label error and outlier/OOD detection tasks with large-scale image, speech, and language datasets: ImageNet [41], ESC-50 [35], and MNLI [53]. Our experiments show state-of-the-art performance on all tasks, demonstrating its effectiveness for debugging and cleaning datasets.

## 2 Related works

**Label error detection** Label errors in datasets can negatively impact model generalization and destabilize evaluation systems [16, 32]. Prior works attempt to address this issue by detecting label errors using bagging and bootstrapping [44, 39]. Other approaches have explored the use of neural networks to learn data sampling schemes or data Shapley values [18, 9]. To mitigate overfitting on label errors, Pleiss et al. [36] propose tracking the training process to measure the area under the margin curve. Recent efforts have shown that simple scoring methods with large pre-trained models, such as prediction margins or loss values, achieve comparable results to previous complex approaches [31, 5]. Meanwhile, Wu et al. [55] propose a unified approach for learning with open-world noisy data. However, this method involves a complicated optimization process during training, which is not suitable for large-scale settings. Another line of approach to identifying label errors involves measuring the influence of a training data point on its own loss [21, 37]. However, these approaches require calculating computationally expensive network gradients on each data point, and their performance is known to be sensitive to outliers and training schemes [2, 3]. In this work, we present a scalable approach that leverages the data relational structure of trained models without necessitating additional training procedures, thereby facilitating the practical analysis of problems in data.

**Outlier/OOD detection** Detecting outlier data is crucial for building robust machine learning systems in real-world environments [22]. A recent survey paper defines the problem of finding outliers in training set as *outlier detection* and finding outliers in the inference process as *OOD detection* [57]. The conventional approach for detecting outliers involves measuring k-nearest distance using efficient sampling methods [46]. More recently, attempts have been made to detect outlier data using scores obtained from trained neural networks, such as Maximum Softmax Probability [14], Energy score [27], and Max Logit score [15]. Other approaches suggest adding perturbations on the inputs or rectifying the activation values to identify the outlier data [26, 47]. Lee et al. [25] propose fitting a Gaussian probabilistic model to estimate the data distribution. Recently, Sun et al. [48] propose a non-parametric approach measuring the  $k$ -nearest feature distance. In our work, we explore the use of the relational structure on the feature-embedded space for identifying outlier data. Our approach is applicable to a wide range of domains without requiring additional training while outperforming existing scoring methods on large-scale outlier/OOD detection benchmarks.

### 3 Methods

In this section, we describe our methodology for identifying label errors and outliers using a model trained on the noisy training dataset. We exploit the feature-embedded structure of the learned neural networks, which are known to effectively capture the underlying semantics of the data [38]. To this end, we first define data relation to construct a data relational graph on the feature space. We then introduce our novel graph algorithms for identifying label errors and outlier data. In Section 3.4, we introduce the data relation map as an effective tool for diagnosing and contextualizing data.

#### 3.1 Data relation

We describe our approach in the context of a classification task, while also noting that the ideas are generalizable to other types of tasks as well. Let us assume we have a trained neural network on a noisy training dataset  $\mathcal{T} = \{(x_i, y_i) \mid i = 1, \dots, n\}$ . For  $x_i \in \mathcal{X}$ , we extract the feature representation  $\mathbf{f}_i$  and the prediction probability vector  $\mathbf{p}_i$  from the model.

We propose a class of bounded kernel  $k : \mathcal{X} \times \mathcal{X} \rightarrow [0, M]$  that measures the semantic similarity between data points on feature space:

$$k(x_i, x_j) = |s(\mathbf{f}_i, \mathbf{f}_j) \cdot c(\mathbf{p}_i, \mathbf{p}_j)|^t. \quad (1)$$

Here,  $t$  is a positive scalar value that controls the sharpness of the kernel value distribution. A larger value of  $t$  makes a small kernel value smaller, which is effective in handling small noisy kernel values. A scalar value  $s(\mathbf{f}_i, \mathbf{f}_j) \in \mathbb{R}^+$  denotes a similarity measurement between features. In our main experiments, we adopt the truncated cosine-similarity that has been widely used in representation learning [42, 40]. We use the hinge function at zero, resulting in the following positive feature-similarity function:

$$s(\mathbf{f}_i, \mathbf{f}_j) = \max(0, \cos(\mathbf{f}_i, \mathbf{f}_j)).$$

It is worth noting the utility of our framework is not limited to a specific kernel design. In Section 4.3, we verify our approach maintains the best performance with  $s(\mathbf{f}_i, \mathbf{f}_j)$  defined as the RBF kernel [58].

While the feature similarity captures the meaningful semantic relationship between data points, we observe that considering the prediction scores  $\mathbf{p}_i$  can further improve the quality of data identification. To incorporate prediction scores into our approach, we introduce a scalar term  $c(\mathbf{p}_i, \mathbf{p}_j)$  that measures the compatibility between the predictions on data points. Any positive and bounded compatibility function is suitable for the kernel class defined in Equation (1). In our main experiments, we use the predicted probability of belonging to the same class as the compatibility term  $c(\mathbf{p}_i, \mathbf{p}_j)$ . Specifically, given the predicted label random variables  $\hat{y}_i$  and  $\hat{y}_j$ , the proposed compatibility term is

$$c(\mathbf{p}_i, \mathbf{p}_j) = P(\hat{y}_i = \hat{y}_j) = \mathbf{p}_i^\top \mathbf{p}_j. \quad (2)$$

From a different perspective, we interpret this compatibility term as a measure of confidence for feature similarity. In Appendix D.4, we examine the effects of different design choices for the compatibility term through empirical studies.

By incorporating the assigned label information with the similarity kernel in Equation (1), we define the relation function  $r : \mathcal{X} \times \mathcal{Y} \times \mathcal{X} \times \mathcal{Y} \rightarrow [-M, M]$ :

$$r((x_i, y_i), (x_j, y_j)) = 1(y_i = y_j) \cdot k(x_i, x_j), \quad (3)$$

where  $1(y_i = y_j) \in \{-1, 1\}$  is a signed indicator value. The relation function reflects the degree to which data samples are complementary or conflicting with each other. In Figure 3, the center image with a label error has negative relations to the left samples that belong to the same ground-truth class. In contrast, the two left samples with correct labels have a positive relation. We also note that samples with dissimilar semantics exhibit near-zero relations.

Our relation function in Equation (3) only requires the forward computation of neural networks, which is embarrassingly parallelizable and scalable to large-scale settings.

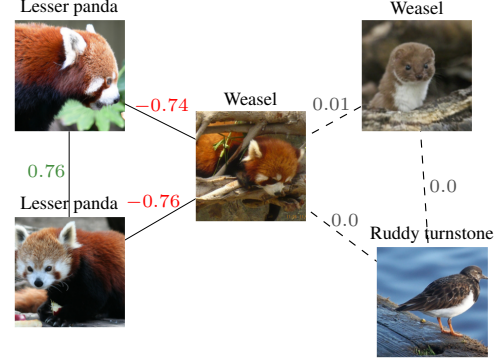


Figure 3: A relation graph with samples from ImageNet and the MAE-Large model [13]. We denote the assigned label above each sample. Here, the center image has a label error.

**Interpretation** To better understand our relation function, we draw a connection to the influence function [37], which estimates the influence between data points by computing the inner product of the network-weight gradient on the loss function  $\ell$  of each data point as  $\nabla_w \ell(x_i)^\top \nabla_w \ell(x_j)$ , where  $w$  denotes the network weights. Following the convention, we consider an influence function on the feed-forward layer, where  $\ell(x_i) = h(\mathbf{f}'_i) = h(w^\top \mathbf{f}_i)$ . By the chain rule, we can decompose the weight gradient as  $\nabla_w \ell(x_i) = \nabla_{\mathbf{f}'} h(\mathbf{f}'_i) \mathbf{f}_i^\top$ , and represent the influence as  $\nabla_{\mathbf{f}'} h(\mathbf{f}'_i)^\top \nabla_{\mathbf{f}'} h(\mathbf{f}'_j) \cdot \mathbf{f}_i^\top \mathbf{f}_j$ . Our relation function differs from the influence function in that it does not rely on feature gradients  $\nabla_{\mathbf{f}'} h(\mathbf{f}'_i)$  to evaluate the relationship between data points. Instead, our relation function compares model predictions and assigned labels independently using the terms  $c(\mathbf{p}_i, \mathbf{p}_j)$  and  $1(y_i, y_j)$ . Our formulation does not require computationally expensive back-propagation and more robustly identifies conflicting data information than influence functions which are known to be sensitive to outliers [2]. We provide a detailed theoretical analysis in Appendix A.4. In the following sections, we present novel graph algorithms for identifying label errors and outliers by utilizing our relation function.

### 3.2 Label error detection

We consider a fully-connected undirected graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ , where the set of nodes  $\mathcal{V}$  corresponds to  $\mathcal{T}$  and the weights  $\mathcal{W}$  on edges  $\mathcal{E}$  are the relation values defined in Equation (3). For clarity, we denote a data point by an index, *i.e.*,  $\mathcal{T} = \{1, \dots, n\}$  and  $r((x_i, y_i), (x_j, y_j)) = r(i, j)$ . Additionally, we set  $r(i, i) = 0$  for  $i \in \mathcal{T}$ . Consistent with previous works [31], we aim to measure the *label noisiness* score for each data, where a *lower* score indicates a *higher* likelihood of label error. We denote the label noisiness scores for  $\mathcal{T}$  as  $s \in \mathbb{R}^n$ , where  $s[i]$  is the score for data  $i$ .

As depicted in Figure 3, data with label errors exhibit negative relations with other samples, implying that the data samples have similar features in the embedding space yet have dissimilarly assigned labels. This indicates that a node’s edge weights measure the degree to which the assigned label aligns with the labels of other nodes. However, simply aggregating all edge weights of a node can yield suboptimal results, as negative edge weights can also contribute to the score for clean data, as shown in Figure 3. In Appendix D.5, we provide a more detailed experimental analysis of this issue.

To rectify this issue, we develop an algorithm that takes into account the global structure of the graph, not simply summing the edge weights of individual nodes. Specifically, we identify subsets of data likely to have correct/incorrect labels and calculate the label noisiness scores based on them. To this end, we partition the nodes into two groups such that the sum of edges between the groups has the lowest negative value, meaning that the label information of the two groups is the most different. Since the label error ratio is typically lower than the half of the training set, we estimate the smaller set to be the noisy set  $\mathcal{N}$  and formulate this as the following min-cut problem:

$$\mathcal{N}^* = \operatorname{argmin}_{\mathcal{N} \subset \mathcal{T}} \operatorname{cut}(\mathcal{N}, \mathcal{T} \setminus \mathcal{N}) \left( := \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{T} \setminus \mathcal{N}} r(i, j) \right), \quad \text{subject to } |\mathcal{N}| < n/2. \quad (4)$$



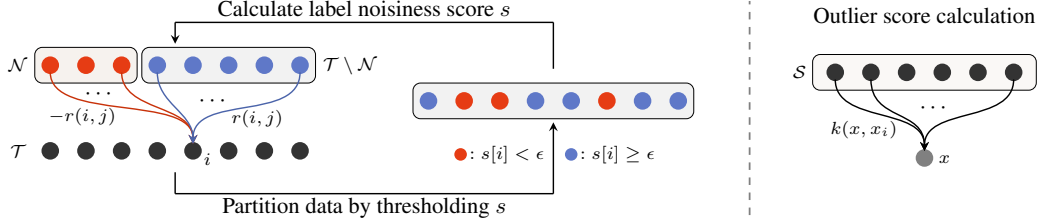


Figure 4: Illustration of our scoring algorithms for identifying label noise (left) and outliers (right).

The min-cut problem with the signed edges is NP-complete [11]. To solve this problem, we adopt the Kernighan-Lin algorithm, which finds a local minimum solution by iteratively updating the solution [19]. However, the original algorithm that swaps data one by one at each optimization iteration is not suitable for large-scale settings. To this end, we propose an efficient *set-level* algorithm in Algorithm 1 that alternatively updates the noisy set  $\mathcal{N}$  and label noisiness score vector  $s$ .

Specifically, given the current estimation of  $\mathcal{N}$ , the cut value excluding edges of node  $i \in \mathcal{T}$  is  $\text{cut}(\mathcal{N} \setminus \{i\}, \mathcal{T} \setminus \mathcal{N} \setminus \{i\})$ . Algorithm 1 measures the label noisiness score of node  $i$  by comparing the objective values when including  $i$  in  $\mathcal{N}$  and when including  $i$  in  $\mathcal{T} \setminus \mathcal{N}$ :

$$s[i] = \text{cut}(\mathcal{N} \cup \{i\}, \mathcal{T} \setminus \mathcal{N} \setminus \{i\}) - \text{cut}(\mathcal{N} \setminus \{i\}, \mathcal{T} \setminus \mathcal{N} \cup \{i\}) = \sum_{j \in \mathcal{T} \setminus \mathcal{N}} r(i, j) - \sum_{j \in \mathcal{N}} r(i, j).$$

Here we use the assumption  $r(i, i) = 0$ . In practice, the number of elements in  $\mathcal{N}$  is small, so we can efficiently update the score vector  $s$  by caching the initial score vector  $\bar{s}$  as in Algorithm 1. After calculating the score vector  $s$ , we update the noisy set  $\mathcal{N}$  by selecting nodes with score values below the threshold  $\epsilon$ . Figure 4 illustrates the optimization process. We can obtain the solution to Equation (4) with  $\epsilon = 0$ , while negative values of  $\epsilon$  result in smaller  $\mathcal{N}$  consisting of data samples that are more likely to have label noise. We provide the sensitivity analysis of  $\epsilon$  in Appendix C.1.

Algorithm 1 satisfies the convergence property in Proposition 1. In Appendix A, we provide proof and present an empirical convergence analysis on large-scale datasets. We also conduct a run-time analysis of our algorithm in Appendix A.3, demonstrating that the computation overhead of Algorithm 1 is negligible in large-scale settings.

**Proposition 1.** *Algorithm 1 with a single node update at each iteration converges to local minima.*

**Complexity analysis** The time complexity of Algorithm 1 is  $O(n^2)$ , proportional to the number of edges in a graph. It is noteworthy that our method maintains the best performance when used with graphs consisting of a small number of nodes, around 12k (Figure 5). This implies that we can partition large datasets and run the algorithm repeatedly for each partition to enhance efficiency while maintaining performance. In this case, the complexity becomes  $O(n/k \cdot k^2) = O(nk)$ , with  $k$  representing the size of each partition and  $n/k$  being the number of partitions. Also, computations on these partitions are embarrassingly parallelizable, meaning that the complexity becomes  $O(k^2)$  for  $k \ll n$  in distributed computing environments.

### 3.3 Outlier/OOD detection

In the previous section, we presented a method for detecting label errors based on data relations with similar feature embeddings but different label information. By employing the identical feature embedding structure, we identify outlier data by measuring the extent to which similar data are absent in the feature embedding space. To quantify the extent of a data point being an outlier, we aggregate the similarity kernel values of a data point in Equation (1), thereby processing the entire relational

---

#### Algorithm 1 Label noise identification

---

**Input:** Relation function  $r$ , threshold value  $\epsilon$

**Notation:** The number of data  $n$

**for**  $i = 1$  **to**  $n$  **do**

    # initialize label noisiness scores

$\bar{s}[i] = \sum_{j=1}^n r(i, j)$

**end for**

$s = \bar{s}$

**repeat**

    # update noisy data subset

$\mathcal{N} = \{i \mid s[i] < \epsilon, i \in [1, \dots, n]\}$

    # update label noisiness scores

**for**  $i = 1$  **to**  $n$  **do**

$s[i] \leftarrow \bar{s}[i] - 2 \sum_{j \in \mathcal{N}} r(i, j)$

**end for**

**until** convergence

**Output:**  $s, \mathcal{N}$

---

information of the data point. Our approach leverages global information about the data distribution, resulting in a more robust performance across a range of experimental settings compared to existing methods that rely on local information such as  $k$ -nearest distance [48]. Specifically, for a subset  $\mathcal{S} \subseteq \mathcal{T}$  and data  $x$ , we measure the outlier score as

$$\text{outlier}(x) = \sum_{i \in \mathcal{S}} k(x, x_i).$$

Lower values in the outlier score indicate that the data are more distributionally outliers. We propose to use a *uniform random* sampling for  $\mathcal{S}$ , adjusting the computational cost and memory requirements for the outlier score calculation to suit the inference environment. In Section 4.2, we verify our method maintains the best OOD detection performance even when using only 0.4% of the data in ImageNet.

### 3.4 Data relation map

Contextualizing data and gaining an understanding of the underlying problems are important for managing data quality. One effective approach is dataset cartography, which projects the dataset onto a 2D plot [49]. Specifically, the approach draws a scatter plot of the mean and standard deviation of the model’s prediction probabilities for each data sample during training. Inspired by the dataset cartography, we propose a novel data contextualization tool, called the data relation map, which visualizes the relationship between data along the training process. To this end, we uniformly store checkpoints during training. We denote a set of these checkpoints as  $\mathcal{K}$ , where  $r_k$  refers to the relation function for checkpoint  $k \in \mathcal{K}$ . For each data sample  $i$ , we draw a scatter plot of the mean and standard deviation of relation values  $\{r_k(i, j) \mid k \in \mathcal{K}\}$  for  $j \in \mathcal{T} \setminus \{i\}$ .

In Figure 2, we provide relation maps of three samples from ImageNet, using 10 checkpoints of MAE-Large [13]. The three samples each represent clean data, data with a label error, and outlier data. From the figure, samples show different relation map patterns. Specifically, the relation map of a clean data sample exhibits a majority of positive relations with relatively small variability. We note that there are gray-colored relations in high variability regions ( $0.2 < \text{std}$ ), indicating that the model resolves conflicting relations at convergence. On the other hand, the relation map of the sample with a label error demonstrates a majority of negative relations. Notably, high variance relations result in largely negative relations at convergence, suggesting that conflicts intensify. Lastly, the relation map of the outlier data sample reveals that relations are close to 0 during training. These relation maps can serve as a model-based fingerprint of the data, which our algorithm effectively exploits to identify problematic data. We provide additional data relational maps for various models in Appendix E.1.

## 4 Experimental results

In this section, we experimentally verify the effectiveness of our approach in detecting label errors and outliers. We provide implementation details including hyperparameter settings in Appendix C.1. We provide qualitative results including detected label errors and outlier samples in Appendix E.2.

### 4.1 Label error detection

#### 4.1.1 Setting

**Datasets** We conduct label error detection experiments on large-scale datasets: ImageNet [41], ESC-50 [35], and MNLI [53]. ImageNet consists of about 1.2M image data from 1,000 classes. ESC-50 consists of 2,000 5-second environmental audio recordings organized into 50 classes. MNLI consists of about 393k sentence pairs with textual entailment annotations. The task is to classify whether the premise sentence entails or contradicts the hypothesis sentence or is neutral. For qualitative analysis, we additionally consider SST2 [53], a binary sentiment classification dataset consisting of 67k movie review sentences.

Following Pruthi et al. [37], we construct a noisy training set by flipping labels of certain percentages of correctly classified training data with the top-2 prediction of the trained model. We use different neural network architectures for constructing a noisy training set and detecting label errors to avoid possible correlation. We leave a more detailed procedure for constructing the noisy training set in Appendix C.2. When training the MAE-Large model [13] on ImageNet with 8% label noise, the validation top1-accuracy decreases by 1.7% compared to the performance of the model trained on the orig-

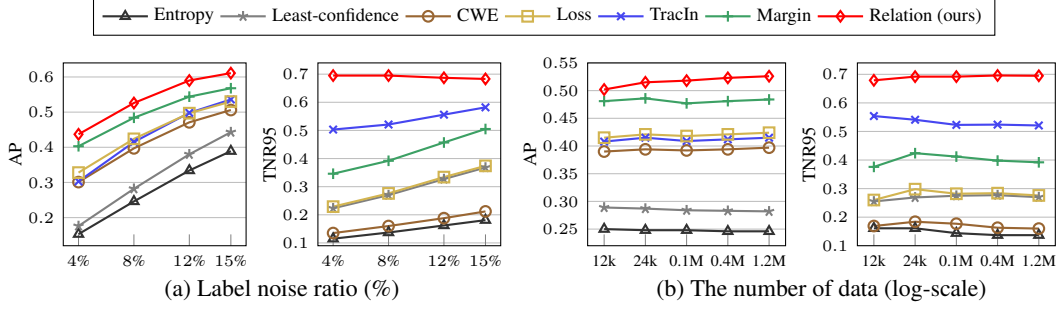


Figure 5: Label error detection performance on ImageNet with MAE-Large according to (a) label noise ratios and (b) the number of data. We obtain the results in (b) with 8% label noise. We report performance values of all methods in Appendix D.1, with Tables 14 and 15.

inal training set, as reported in Appendix Table 8. The decrease becomes more significant with 15% label noise, with a drop of 4.4%, highlighting the importance of label noise detection and cleaning.

**Baselines** We compare our method (*Relation*) to six baselines that are suitable for large-scale datasets. We consider fine-tuned loss from pre-trained models (*Loss*) [5], prediction probability margin score (*Margin*) [31], and the influence-based approach called *TracIn* [37]. We also evaluate model-agnostic scoring methods: *Entropy*, *Least-confidence*, and Confidence-weighted Entropy (*CWE*) [23]. For a fair comparison, we evaluate methods using a single converged neural network in our main experiments, while also providing results with a temporal ensemble suggested by [37] in Appendix D.3.

**Metric** We evaluate the detection performance based on label noisiness scores by each method. We note that detecting label errors is an imbalanced detection problem, which makes the AUROC metric prone to being optimistic and misleading [7]. In this respect, we mainly report the AP (average precision) and TNR95 (TNR at 0.95 TPR), and provide AUROC results in Appendix D.1.

#### 4.1.2 Results and analysis

**ImageNet** We measure the label error detection performance on ImageNet with the synthetic label noise by training an MAE-Large model [13]. Note that the model does not have access to information about the changed clean labels during the entire training process. Figure 5 (a) shows the detection performance over a wide range of label noise ratios from 4% to 15%. As shown in the figure, our approach achieves the best AP and TNR95 performance compared to the baselines. Especially, our method maintains a high TNR95 over a wide range of noise ratios, indicating that the number of data that need to be reviewed by human annotators is significantly smaller when cleaning the dataset. In Figure 9, we present detected label error samples by our algorithm.

It is worth noting that our method relies on the number of data for constructing a relation graph. To measure the sensitivity of our algorithm to the number of data, we evaluate the detection performance using a reduced number of data with uniform random sampling. Figure 5 (b) shows the detection performance on 8% label noise with MAE-Large. From the figure, we find that our algorithm maintains the best detection performance even with 1% of the data (12k). This demonstrates that our algorithm is effective even when only a small portion of the training data is available, such as continual learning or federated learning [33, 30]. In Table 1 (a), we provide detection performance for different scales of MAE models on 8% label noise. The table shows our approach achieves the best AP with MAE-Base, verifying the robustness of our approach to the network scales. From the table, we note that larger models are more robust to label noise and show better detection performance.

**Speech and language domains** We apply our method to speech and language domain datasets: ESC-50 [35] and MNLI [53]. We design the label noise detection settings identical to the previous ImageNet section. Specifically, we train the AST model [10] for ESC-50 and the RoBERTa-Base model [28] for MNLI under the 10% label noise setting. Table 1 (b) shows our approach achieves the best AP and TNR95 on the speech and language datasets, demonstrating the generality of our approach across various data types.

Table 1: Label error detection performance on ImageNet with 8% label noise. *Baseline* refers to the *best* performance among the six baselines considered in Figure 5. In Table (c), the evaluation metric is AP. We report the performance of all baselines in Appendix D.1, with Tables 9 to 11.

(a) Model architecture scales				(b) Speech/language domains				(c) Realistic label noise scenario		
Scale	Metric	Baseline	Relation	Dataset	Metric	Baseline	Relation	Model	Baseline	Relation
Base	AP	0.477	<b>0.514</b>	ESC50	AP	0.739	<b>0.779</b>	MAE	0.708	<b>0.733</b>
	TNR95	0.488	<b>0.672</b>		TNR95	0.793	<b>0.847</b>	BEIT	0.719	<b>0.737</b>
Large	AP	0.484	<b>0.526</b>	MNLI	AP	0.764	<b>0.766</b>	ConvNeXt	0.713	<b>0.735</b>
	TNR95	0.521	<b>0.695</b>		TNR95	0.514	<b>0.660</b>	ConvNeXt-22k	0.724	<b>0.744</b>

**Realistic label noise** The ImageNet validation set is known to contain numerous label errors [32]. To tackle this issue, Beyer et al. [4] cleaned the labels with human experts and corrected around 29% of the labels via multi-labeling. With this re-labeled validation set, we conduct experiments under the realistic label noise, with the task of detecting the data samples with changed labels. We measure the detection performance with MAE-Large [13], BEIT-Large [1], and ConvNeXt-Large [29] models. To examine the impact of pre-training on external data, we also include ConvNeXt pre-trained on ImageNet-22k, denoted as ConvNeXt-22k. We construct the relation graph using only the validation set, considering scenarios where the training data are not available. Table 1 (c) verifies that our approach outperforms the best baseline across various models. The results on ConvNeXt-22k indicate that pre-training on external data improves the detection performance.

**Memorization issue** We investigate the impact of large neural networks’ ability to memorize label errors on detection performance [59]. In the left figure of Figure 6, we find that the AP score decreases as the training progresses after 30 epochs with MAE-Large which converges at 50 epochs. The right figure of Figure 6 plots the precision-recall curves, where we observe that precision increases at low recall area but decreases at mid-level recall ( $\sim 0.5$ ) as the training progresses. This suggests that training has both positive and negative effects on detecting label noise, and we speculate that memorization is one cause. Leveraging these observations, we improve detection AP by 3.6%p by using a temporal ensemble of models [24]. We provide a more detailed description and results in Appendix D.3.

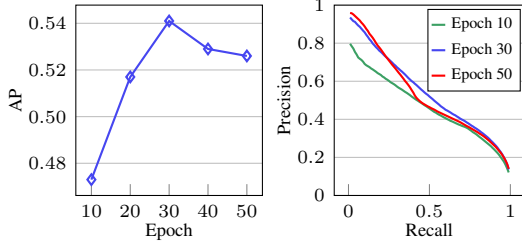


Figure 6: Label error detection performance of relation graph throughout the MAE-Large training process on ImageNet with 8% label noise.

## 4.2 Outlier/OOD detection

**Baselines** We consider the following representative outlier scoring approaches: Maximum Softmax Probability (*MSP*) [14], *Max Logit* [15], *Mahalanobis* [25], *Energy* score [27], *ReAct* [47], *KL-Matching* [15], and *KNN* [48]. We tune the hyperparameter  $k$  of the KNN method following the guideline provided in the paper. We also evaluate famous outlier detection approaches, *Iterative sampling* [46] and *Local outlier factor* [54].

**OOD detection** Following Sun et al. [48], we evaluate OOD detection performance on the ImageNet validation set consisting of 50k in-distribution data samples, along with four distinct OOD datasets: *Places* [60], *SUN* [56], *iNaturalist* [51], and *Textures* [6]. Each of these OOD datasets consists of 10k data samples except for Textures which has 5,640 data samples. We also combine these four datasets, denoted as *ALL*, and measure the overall OOD detection performance on this dataset.

Figure 7 shows OOD detection performance of MAE-Large on ALL outlier dataset. Note that our approach and KNN both rely on the number of training data samples ( $|S|$ ) for outlier score calculation. We examine the effect of training set size by measuring the performance with a reduced number of data using uniform random sampling. Figure 7 verifies that our approach outperforms other baselines while maintaining performance even with 0.4% of the training dataset (5k). Note that KNN requires hyperparameter tuning according to the training set size, whereas our approach uses the identical

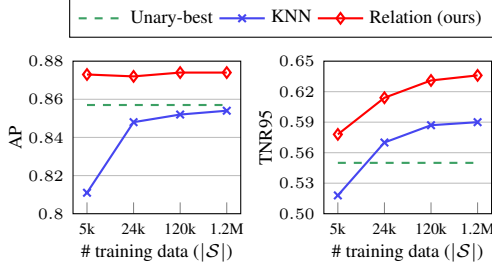


Figure 7: OOD detection performance on ImageNet (ALL) with MAE-Large. *Unary-best* refers to the best performance among the methods that do not rely on the training data for outlier score calculation. We provide performance values of all baselines in Appendix D.2, with Table 16.

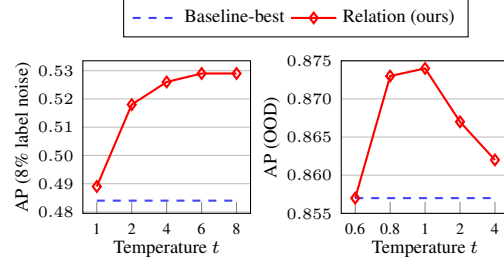


Figure 8: The detection AP of the MAE-Large model across a range of kernel temperatures  $t$ . The dashed blue line means the performance of the best baseline.

Table 2: Outlier detection performance with Vit-Base on noisy ImageNet-100. Some OOD scoring methods (Mahalanobis, ReAct, KL-Matching) are excluded from the comparison because they require a clean training dataset which is not available in the outlier detection setup.

(a) ImageNet-100 with SUN				(b) ImageNet-100 with iNaturalist			
Method	AUROC	AP	TNR95	Method	AUROC	AP	TNR95
MSP	0.708	0.335	0.032	MSP	0.706	0.309	0.040
Max Logit	0.499	0.216	0.011	Max Logit	0.469	0.171	0.012
Energy	0.417	0.106	0.010	Energy	0.375	0.075	0.012
KNN	0.990	0.899	0.960	KNN	0.993	0.923	0.972
Iterative sampling	0.973	0.687	0.903	Iterative sampling	0.979	0.734	0.922
Local outlier factor	0.986	0.850	0.941	Local outlier factor	0.990	0.890	0.958
Relation (Ours)	<b>0.993</b>	<b>0.906</b>	<b>0.971</b>	Relation (Ours)	<b>0.995</b>	<b>0.940</b>	<b>0.982</b>

hyperparameter ( $t = 1$ ) regardless of the size. In Appendix D.2, Tables 16 and 17, we provide OOD detection results on four individual OOD datasets as well as the performance with ResNet-50 [12], where our approach achieves the best OOD detection performance over the nine baselines considered.

**Outlier detection** We perform outlier detection experiments following the methodology by Wang et al. [54], where the training set contains outlier data with random labels. We construct the noisy ImageNet-100 training sets by using iNaturalist [51] and SUN [56] datasets. We train a ViT-Base model [8] from scratch on these noisy training datasets, and measure outlier detection performance using the trained model. For a more detailed description, please refer to Appendix C.3.

Table 2 shows the outlier detection results on two outlier datasets. As indicated, our method achieves the best performance in both outlier settings, demonstrating its effectiveness in outlier detection. It is worth noting that the considered OOD scoring methods (MSP, Max Logit, Energy) do not achieve good outlier detection performance. We speculate that this is due to the overfitting of the neural network’s predictions on outliers.

**Detecting outliers in validation set** We further utilize our method for identifying outliers in the validation set by retrieving data samples with the lowest outlier score (Section 3.3). In Figure 10, we present samples detected by our algorithm from ImageNet and SST2. In the figure, we observe that these samples are not suitable for measuring the predictive performance on labels, which should be excluded from the evaluation dataset.

### 4.3 Ablation study

**Temperature  $t$**  In Equation (1), we introduced a temperature  $t$ , where a large value of  $t$  increases the influence of large relation values in our algorithm. We conduct sensitivity analysis on  $t$  with MAE-Large on ImageNet under 8% label noise. Figure 8 shows the effect of the temperature value

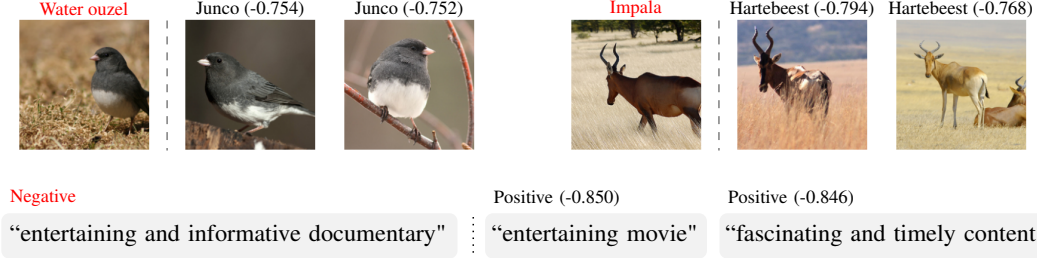


Figure 9: Detected data samples with label errors (marked in red) from ImageNet (top) and SST2 (bottom). We present samples with conflicting relations next to the detected samples and denote the corresponding relation value in parenthesis. We present more samples in Appendix E.2, Figure 14.

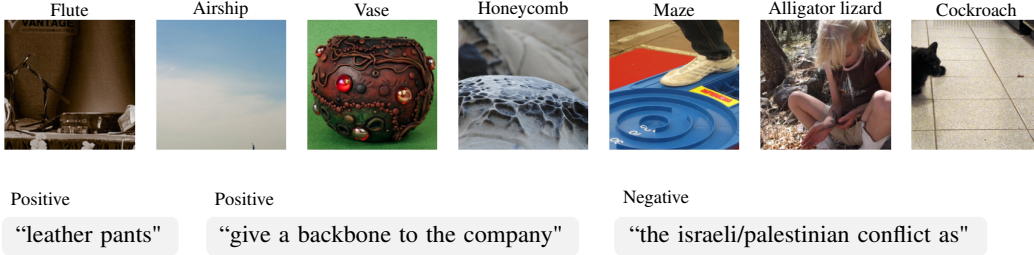


Figure 10: Data samples with the lowest outlier scores by our method on ImageNet (top) and SST2 (bottom) validation sets. We denote the assigned labels above each data sample. We present more outlier samples in Appendix E.2, Figure 15.

on our detection algorithm’s performance. From the figure, we observe that the label error detection performance increases as the  $t$  value increases, saturating at  $t = 6$ . In the case of OOD detection, we achieve the best performance at around  $t = 1$ . Our algorithm outperforms the best baseline over a wide range of hyperparameters, demonstrating the robustness of our algorithm to the hyperparameter.

**Similarity kernel design** We present an empirical analysis of the kernel design choices. Specifically, we replace the cosine similarity term in Equation (1) as the RBF kernel and evaluate the detection performance. We further conduct an ablation study on compatibility terms (Equation (2)). Table 3 summarizes the label error detection performance with different kernel functions on ImageNet with 8% noise ratio. The table shows that our approach largely outperforms the best baseline even with the RBF

kernel. Also, we find that our approach without the compatibility term shows comparable AP performance while significantly outperforming baselines in TNR95. These results demonstrate the generality and utility of our relational structure-based framework, which is not limited to a specific kernel design.

Table 3: Comparison of similarity kernel designs. *Baseline* represents the best baseline performance. The term  $c$  denotes our compatibility term in Equation (2). Note,  $Cos \cdot c$  is the kernel function considered in our main experiments, and  $RBF / Cos$  refers to our method without the compatibility term  $c$ .

Metric	Baseline	RBF	Cos	RBF $\cdot c$	Cos $\cdot c$
AP	0.484	0.470	0.471	0.525	0.526
TNR95	0.521	0.668	0.671	0.703	0.695

## 5 Conclusion

In this paper, we propose a novel data relation function and graph algorithms for detecting label errors and outlier data using the relational structure of data in the feature embedding space. Our approach achieves state-of-the-art performance in both label error and outlier/OOD detection tasks, as demonstrated through extensive experiments on large-scale benchmarks. Furthermore, we introduce a data contextualization tool based on our data relation that can aid in data diagnosis. Our algorithms and tools can facilitate the analysis of large-scale datasets, which is crucial for the development of robust machine-learning systems.



## References

- [1] H. Bao, L. Dong, and F. Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2022.
- [2] E. Barshan, M.-E. Brunet, and G. K. Dziugaite. Relatif: Identifying explanatory training samples via relative influence. In *AISTATS*, 2020.
- [3] S. Basu, P. Pope, and S. Feizi. Influence functions in deep learning are fragile. In *ICLR*, 2021.
- [4] L. Beyer, O. J. Hénaff, A. Kolesnikov, X. Zhai, and A. v. d. Oord. Are we done with imagenet? *arXiv preprint arXiv:2006.07159*, 2020.
- [5] D. Chong, J. Hong, and C. D. Manning. Detecting label errors using pre-trained language models. *arXiv preprint arXiv:2205.12702*, 2022.
- [6] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *CVPR*, 2014.
- [7] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *ICML*, 2006.
- [8] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.
- [9] A. Ghorbani and J. Zou. Data shapley: Equitable valuation of data for machine learning. In *ICML*, 2019.
- [10] Y. Gong, Y.-A. Chung, and J. Glass. Ast: Audio spectrogram transformer. In *Interspeech*, 2021.
- [11] J. Hartmanis. Computers and intractability: a guide to the theory of np-completeness (michael r. Garey and david s. Johnson). *Siam Review*, 24(1), 1982.
- [12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- [13] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.
- [14] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.
- [15] D. Hendrycks, S. Basart, M. Mazeika, M. Mostajabi, J. Steinhardt, and D. Song. Scaling out-of-distribution detection for real-world settings. In *ICML*, 2022.
- [16] W. Hu, Z. Li, and D. Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *ICLR*, 2021.
- [17] A. Ilyas, S. M. Park, L. Engstrom, G. Leclerc, and A. Madry. Datamodels: Predicting predictions from training data. In *ICML*, 2022.
- [18] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, 2018.
- [19] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell system technical journal*, 49(2), 1970.
- [20] J.-H. Kim, J. Kim, S. J. Oh, S. Yun, H. Song, J. Jeong, J.-W. Ha, and H. O. Song. Dataset condensation via efficient synthetic-data parameterization. In *ICML*, 2022.
- [21] P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In *ICML*, 2017.
- [22] P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *ICML*, 2021.
- [23] J. Kuan and J. Mueller. Model-agnostic label quality scoring to detect real-world label errors. In *ICML DataPerf Workshop*, 2022.
- [24] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. In *ICLR*, 2017.
- [25] K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018.



- [26] S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.
- [27] W. Liu, X. Wang, J. Owens, and Y. Li. Energy-based out-of-distribution detection. In *NeurIPS*, 2020.
- [28] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [29] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. In *CVPR*, 2022.
- [30] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
- [31] C. Northcutt, L. Jiang, and I. Chuang. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 2021.
- [32] C. G. Northcutt, A. Athalye, and J. Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. In *NeurIPS Datasets and Benchmarks Track*, 2021.
- [33] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113, 2019.
- [34] W. Park, D. Kim, Y. Lu, and M. Cho. Relational knowledge distillation. In *CVPR*, 2019.
- [35] K. J. Piczak. ESC: Dataset for Environmental Sound Classification. In *Proceedings of the 23rd Annual ACM Conference on Multimedia*, 2015.
- [36] G. Pleiss, T. Zhang, E. Elenberg, and K. Q. Weinberger. Identifying mislabeled data using the area under the margin ranking. In *NeurIPS*, 2020.
- [37] G. Pruthi, F. Liu, S. Kale, and M. Sundararajan. Estimating training data influence by tracing gradient descent. In *NeurIPS*, 2020.
- [38] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.
- [39] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.
- [40] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP-IJCNLP*, 2019.
- [41] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3), 2015.
- [42] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.
- [43] K. Shuster, J. Xu, M. Komeili, D. Ju, E. M. Smith, S. Roller, M. Ung, M. Chen, K. Arora, J. Lane, et al. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*, 2022.
- [44] B. Sluban, D. Gamberger, and N. Lavrač. Ensemble-based noise detection: noise ranking and visual performance evaluation. *Data mining and knowledge discovery*, 28(2), 2014.
- [45] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016.
- [46] M. Sugiyama and K. Borgwardt. Rapid distance-based outlier detection via sampling. In *NeurIPS*, 2013.
- [47] Y. Sun, C. Guo, and Y. Li. React: Out-of-distribution detection with rectified activations. In *NeurIPS*, 2021.
- [48] Y. Sun, Y. Ming, X. Zhu, and Y. Li. Out-of-distribution detection with deep nearest neighbors. In *ICML*, 2022.
- [49] S. Swayamdipta, R. Schwartz, N. Lourie, Y. Wang, H. Hajishirzi, N. A. Smith, and Y. Choi. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *EMNLP*, 2020.

- [50] M. Toneva, A. Sordoni, R. T. d. Combes, A. Trischler, Y. Bengio, and G. J. Gordon. An empirical study of example forgetting during deep neural network learning. In *ICLR*, 2019.
- [51] G. Van Horn, O. Mac Aodha, Y. Song, Y. Cui, C. Sun, A. Shepard, H. Adam, P. Perona, and S. Belongie. The inaturalist species classification and detection dataset. In *CVPR*, 2018.
- [52] V. Vasudevan, B. Caine, R. Gontijo-Lopes, S. Fridovich-Keil, and R. Roelofs. When does dough become a bagel? analyzing the remaining mistakes on imagenet. In *NeurIPS*, 2022.
- [53] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*, 2019.
- [54] Y. Wang, W. Liu, X. Ma, J. Bailey, H. Zha, L. Song, and S.-T. Xia. Iterative learning with open-set noisy labels. In *CVPR*, 2018.
- [55] Z.-F. Wu, T. Wei, J. Jiang, C. Mao, M. Tang, and Y.-F. Li. Ngc: A unified framework for learning with open-world noisy data. In *ICCV*, 2021.
- [56] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.
- [57] J. Yang, K. Zhou, Y. Li, and Z. Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021.
- [58] B. Yekkehkhany, A. Safari, S. Homayouni, and M. Hasanlou. A comparison study of different kernel functions for svm-based classification of multi-temporal polarimetry sar data. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2014.
- [59] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3), 2021.
- [60] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6), 2017.

## A Algorithm analysis

### A.1 Proof

In this section, we provide proof for Proposition 1. Recall that Algorithm 1 updates an estimated noisy set  $\mathcal{N}$  at a set-level as  $\mathcal{N} = \{i \mid s[i] < \epsilon\}$ . We can also conduct Algorithm 1 with a single node update by moving one sample at each iteration, referred to as a sample-level version of Algorithm 1. Specifically, let  $v \in \mathbb{R}^n$  such that  $v[i] = -1$  for  $i \in \mathcal{N}$  and  $v[i] = 1$  for else. Then we move a sample  $k$  to another partition at each iteration, where  $k = \operatorname{argmin}_{i \in \mathcal{T}} v[i]s[i]$ . The algorithm stops when  $0 \leq v[k]s[k]$ .

**Proposition 1.** *Algorithm 1 with a single node update at each iteration converges to local minima.*

*Proof.* The change in the objective value of Equation (4) by moving data  $i$  from  $\mathcal{T} \setminus \mathcal{N}$  to  $\mathcal{N}$  is

$$\sum_{j \in \mathcal{T} \setminus \mathcal{N}} r(i, j) - \sum_{j \in \mathcal{N}} r(i, j), \quad (5)$$

where the change by moving data  $i$  from  $\mathcal{N}$  to  $\mathcal{T} \setminus \mathcal{N}$  is

$$\sum_{j \in \mathcal{N}} r(i, j) - \sum_{j \in \mathcal{T} \setminus \mathcal{N}} r(i, j).$$

Algorithm 1 updates the score  $s$  by Equation (5) as

$$s[i] = \sum_{j \in \mathcal{T}} r(i, j) - 2 \sum_{j \in \mathcal{N}} r(i, j) = \sum_{j \in \mathcal{T} \setminus \mathcal{N}} r(i, j) - \sum_{j \in \mathcal{N}} r(i, j).$$

The change in the objective value by moving data  $i$  to another partition becomes  $s[i]$  for  $i \in \mathcal{T} \setminus \mathcal{N}$  and  $-s[i]$  for  $i \in \mathcal{N}$ , which is  $v[i]s[i]$ . Therefore, moving a sample with a negative value of  $v[i]s[i]$  to another partition guarantees a decrease in the objective function value. Because a cut value of a graph is bounded, the algorithm converges to local minima by the monotone convergence theorem.  $\square$

### A.2 Empirical convergence analysis

We conduct an empirical study on the convergence of Algorithm 1. Specifically, we randomly sample 100,000 data from ImageNet and construct a relation graph. We compare the set-level Algorithm 1 and the original Kerningham-Lin algorithm in Figure 11. The figure indicates that both algorithms converge to local minima, while our set-level algorithm converges faster. Additionally, we observe that the set-level algorithm achieves a lower objective value, verifying its effectiveness in large-scale settings.

### A.3 Computation time comparison

In this section, we measure the time spent on detection algorithms. We use 1 RTX3090-Ti GPU and conduct experiments on the full ImageNet training set. Table 4 compares computation time for Algorithm 1 and feature calculation. Note that all existing methods based on neural networks, including ours, require the calculation of features. The table shows that Algorithm 1 (excluding feature calculation) requires significantly less computation time than forward computation. We also

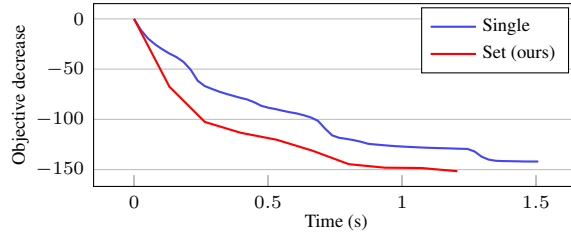


Figure 11: Empirical convergence analysis of min-cut algorithms. *Set* denotes Algorithm 1 and *Single* denotes the Kerningham-Lin algorithm which updates a single node at each optimization step.

Table 4: Time spent (s) for label error detection (ImageNet, MAE-Large). *Feature* indicates the total computing time for calculating feature embeddings of a dataset, and *Gradient* means the total computing time for calculating network gradient on data. *Algorithm 1* indicates the time spent by our algorithm excluding feature calculation.

Model	Feature	Algorithm 1	Gradient
MAE-Base	2300	400	6000
MAE-Large	6900	420	21000

Table 5: Time spent per sample (ms) for OOD detection (ImageNet, MAE-Large). *Unary* refers to methods using logit or probability scores.

Model	Unary	Relation
MAE-Large	12.2	12.3
ResNet-50	8.1	8.2

observe that our algorithm efficiently scales up to larger neural networks which have a larger number of feature embedding dimensions. It is also worth noting that computing gradient takes a much longer time and also requires a large memory budget, demonstrating the efficiency of our algorithm in large-scale label error detection.

In Table 5, we measure the time spent for OOD detection on the full ImageNet training set. Computing our kernel similarity is embarrassingly parallelizable on GPUs. As shown in the table, the overhead time for computing our outlier scores is negligible compared to the time spent for the neural network forward pass on a single data point. Note that we can further reduce the time cost and memory requirements by measuring the outlier score on a subset of the training set as shown in Figure 7.

#### A.4 Interpretation of relation function

We establish an understanding of our relation function in Equation (3) by drawing a connection to the influence function [37]. For simplicity, we consider the influence function with a single checkpoint, where the influence between data  $x_i$  and  $x_j$  is given by  $\nabla_w \ell(x_i)^\top \nabla_w \ell(x_j)$ . Here,  $\ell$  denotes the loss function, and  $w$  denotes the weight of the checkpoint. We consider the influence function at the feed-forward layer, where  $\ell(x_i) = h(\mathbf{f}'_i) = h(w^\top \mathbf{f}_i)$ , following the convention [37]. By the chain rule, we can decompose the weight gradient as  $\nabla_w \ell(x_i) = \nabla_{\mathbf{f}'} h(\mathbf{f}'_i) \mathbf{f}_i^\top$ , and represent the influence as  $\nabla_{\mathbf{f}'} h(\mathbf{f}'_i)^\top \nabla_{\mathbf{f}'} h(\mathbf{f}'_j) \cdot \mathbf{f}_i^\top \mathbf{f}_j$ . In contrast, our relation function has a form of  $1(y_i = y_j) \cdot |s(\mathbf{f}_i, \mathbf{f}_j) \cdot c(\mathbf{p}_i, \mathbf{p}_j)|^t$ .

The main distinction between our relation function and the influence function is the existence of the feature gradient term  $\nabla_{\mathbf{f}'} h(\mathbf{f}'_i)$ . As observed in Barshan et al. [2], outliers have a large feature-gradient norm, leading to difficulties in detecting label errors. Specifically, let us consider the weight  $w$  at the classifier layer, where the function  $h$  is the softmax cross-entropy loss function. As Pruthi et al. [37], we can express the feature gradient inner-product as

$$\nabla_{\mathbf{f}'} h(\mathbf{f}'_i)^\top \nabla_{\mathbf{f}'} h(\mathbf{f}'_j) = (\mathbf{y}_i - \mathbf{p}_i)^\top (\mathbf{y}_j - \mathbf{p}_j),$$

where  $\mathbf{y}_i$  denote the one-hot label. The equation above shows that the correctly classified data with  $\mathbf{y}_i \approx \mathbf{p}_i$  yields near zero inner-product values, whereas outliers with high entropy predictions exhibit large inner-product values. Consequently, existing influence-based label error detection methods, which detect label errors by identifying data with high influence values, have degraded performance in the presence of outliers [2].

Our relation function differs from influence functions in that it separates label and prediction information using a label comparison term  $1(y_i = y_j)$  and a compatibility term  $c(\mathbf{p}_i, \mathbf{p}_j)$ , respectively. Outlier data typically have a high entropy of model predictions, resulting in lower compatibility values with other data [14]. On the other hand, normal data with label errors exhibit high compatibility values with other normal data. Our detection algorithms exploit these differences and achieve improved detection performance compared to the influence functions.

## B Additional discussions

**Why does relation graph work?** We discuss the conceptual differences between our relation graph-based approach and previous baselines. Firstly, our method is data-centric, whereas the previous approaches rely on a unary score by models. Our approach identifies problematic data by

Table 6: Hyperparameter settings. We use identical hyperparameters for all experiments of each task regardless of model types.

Task	$t$ (Equation (1))	$\epsilon$ (Algorithm 1)
Label error	4	-0.05
Outlier	6	-
OOD	1	-

Table 7: Label error detection performance over a range of  $\epsilon$  values (ImageNet 8% noise, MAE-Large).

Metric \ $\epsilon$	-0.1	-0.05	0.0	0.05
AP	0.527	0.527	0.522	0.509
TNR95	0.692	0.695	0.696	0.690

comparing them to other data, which leads to more reliable identification of problematic data than unary scoring methods that are vulnerable to overfitting [36]. Secondly, our approach aggregates global relational information, whereas previous methods rely on local information such as  $k$ -nearest distance [48]. Considering all edge connections, our method obtains more representative information about the data distribution. Through temperature parameter  $t$  and efficient graph algorithms, we effectively process the entire relations and achieve the improved identification of problematic data.

**Limitations and future works** There are several promising future directions for our work. Firstly, the current experiments are limited to the classification task, and it would be valuable to apply our approach to a wider range of tasks, such as segmentation or generative models. These tasks may introduce new and interesting categories of problematic data arising from different label spaces and data structures. Secondly, integrating our method with human annotation and model training processes will also be valuable. This could involve using our approach to identify inconsistencies in label assignments or to conduct a fine-grained evaluation of models.

**Broader impact** Regarding social impact, we anticipate that our method will largely reduce the cost of the data cleaning and annotation process, while also providing developers with valuable insights into their data and training process through our data relation map. We do not anticipate negative social effects, as our method does not deal with socially sensitive issues.

## C Experiment settings

### C.1 Implementation details

**Models** For label error detection, we train models on datasets with label noise. In the case of ImageNet, we fine-tune the pre-trained MAE models following the official training codes<sup>1</sup>, which train MAE-Large for 50 epochs and MAE-Base for 100 epochs. It is worth noting that the masked auto-encoding pre-training process of MAE does not utilize label information. We use 8 RTX3090-Ti GPUs for the training and 1 RTX3090-Ti GPU for executing our algorithm. In the case of ESC-50, we fine-tune the AST model for 25 epochs following the official training codes<sup>2</sup>. In the case of language domain tasks, we fine-tune RoBERTa-Base following the official training codes<sup>3</sup>, where we train models for 5 epochs. For other models, we use the trained models provided by the Timm library<sup>4</sup>. For all experiments, we use the inputs of the classification layers as feature embeddings. In the case of RoBERTa, this corresponds to the encoder output of the [CLS] token.

**Hyperparameter** As shown in Figure 8, we observe that a large value of temperature  $t$  benefits label error detection, while a moderate temperature value around 1 shows the best performance on OOD detection. We summarized the hyperparameter used in our experiments in Table 6. Note we use  $\epsilon = -0.05$  in Algorithm 1 after scaling the label noisiness score to have a maximum absolute value of 1. We observe that the conservative estimation of noisy set  $\mathcal{N}$  by using small negative  $\epsilon$  values leads to slightly better results than  $\epsilon = 0$ , as shown in Table 7. The results in the table confirm that our method performs effectively over a wide range of  $\epsilon$  values, outperforming the best scores of 0.484 AP and 0.521 TNR95 from the six baselines tested.

<sup>1</sup><https://github.com/facebookresearch/mae>

<sup>2</sup><https://github.com/YuanGongND/ast>

<sup>3</sup><https://github.com/facebookresearch/fairseq/tree/main/examples/roberta>

<sup>4</sup><https://github.com/rwightman/pytorch-image-models>

Table 8: Validation top-1 accuracy of MAE-Large trained on ImageNet with noisy labels.

Label Noise Ratio	0.	0.04	0.08	0.12	0.15
Top-1 Accuracy	85.89	84.96	84.15	82.88	81.50

Table 9: Label error detection performance on various scales of MAE (ImageNet, 8% noise).

Scale	Metric	Entropy	Least-conf.	CWE	Loss	TracIn	Margin	Relation
Base	AP	0.231	0.280	0.373	0.412	0.393	0.477	<b>0.514</b>
	TNR95	0.12	0.217	0.132	0.223	0.488	0.342	<b>0.672</b>
Large	AP	0.246	0.282	0.397	0.424	0.415	0.484	<b>0.526</b>
	TNR95	0.137	0.270	0.160	0.276	0.521	0.392	<b>0.695</b>

**Other tricks** We find that small noisy kernel values accumulate errors as we consider large numbers of data. To resolve this issue, we clamp small similarity kernel values that fall below an absolute value of 0.03 as zero in Equation (1).

## C.2 Synthetic label noise

In Section 4.1, we conduct controlled experiments by generating synthetic label noise. Specifically, we flip labels of a certain percentage of training data with the top-2 prediction of trained models on correctly classified data. For the label flip, we use MAE-Huge for ImageNet and RoBERTa-Large for language datasets. Note that we did not use these models for detecting label errors to prevent possible correlations. In the case of speech domain, we use the identical AST architecture. We note that the original ImageNet training set may contain label issues which can lead to misleading experimental results [32]. Therefore, we remove about 4% of data that are likely to have label issues by following Northcutt et al. [31] with MAE-Huge, resulting in a total of 1,242,890 data samples. We conduct label error detection experiments on this pre-cleaned training set. In Table 8, we provide the top-1 validation accuracy of MAE-Large trained on training sets with noisy labels, demonstrating the importance of label noise identification and cleaning.

## C.3 Outlier detection

As outlier detection is not well-benchmarked in modern computer vision [57], we design an experimental setup with the ImageNet dataset and two outlier datasets: SUN [56] and iNaturalist [51], each with 10k outlier data. To ensure an appropriate outlier data ratio ( $\sim 8\%$ ), we use ImageNet-100 [20], a subset of ImageNet consisting of 120k data from 100 classes. We adopt the outlier detection setting of Wang et al. [54], where the training set has the outlier data with random labels. To consider multiple types of outlier data, we construct two noisy training sets: ImageNet-100 with SUN and ImageNet-100 with iNaturalist. We train a ViT-Base model [8] from scratch for 300 epochs on these noisy training datasets and measure outlier detection performance using the trained model.

# D Additional experimental results

## D.1 Label error detection

In this section, we provide the exact performance values for Figure 5, including AUROC results. We report label error detection performances under various noise levels in Table 14, and provide performances according to the number of data in Table 15. The tables confirm that our relation graph approach achieves the best label error detection performances in all three metrics, regardless of the noise ratio and the number of data. In Tables 9 to 11, we provide performance values of all baselines considered in Table 1.

Table 10: Label error detection performance on speech (ESC-50) and language (MNLI) datasets with 10% label noise. For the ESC-50, we use the AST model [10], and for the MNLI, we use the RoBERTa-Base model [28].

Dataset	Metric	Entropy	Least-conf.	CWE	Loss	TracIn	Margin	Relation
ESC-50	AP	0.715	0.737	0.720	0.737	0.739	0.737	<b>0.779</b>
	TNR95	0.784	0.783	0.790	0.783	0.789	0.793	<b>0.847</b>
MNLI	AP	0.199	0.197	0.753	0.764	0.724	0.754	<b>0.766</b>
	TNR95	0.103	0.104	0.494	0.497	0.510	0.514	<b>0.660</b>

Table 11: Label error detection AP on ImageNet validation set. All the model scales are Large.

Model	Entropy	Least-conf.	CWE	Loss	TracIn	Margin	Relation
MAE	0.558	0.609	0.688	0.703	0.695	0.708	<b>0.733</b>
BEIT	0.614	0.644	0.707	0.719	0.718	0.718	<b>0.737</b>
ConvNeXt	0.587	0.625	0.696	0.710	0.700	0.713	<b>0.735</b>
ConvNeXt-22k	0.617	0.642	0.707	0.722	0.719	0.724	<b>0.744</b>

Table 12: Label error detection AP by the temporal model ensemble with 4 MAE-Large checkpoints (ImageNet, 8% label noise). In parenthesis, we denote the performance gain compared to the detection by a single converged model.

Entropy	Least-conf.	CWE	Loss	TracIn	Margin	Relation
0.246 (0.007)	0.282 (0.001)	0.397 (0.031)	0.465 (0.041)	0.449 (0.034)	0.544 (0.06)	<b>0.562</b> (0.036)

## D.2 OOD detection

In Tables 16 and 17, we provide OOD detection results on individual datasets mentioned in Section 4.2: Places, SUN, iNaturalist, and Textures. We report the OOD detection performance of MAE-Large in Table 16 and the performance of ResNet-50 in Table 17. The tables demonstrate that our approach achieves the best OOD detection performance on three out of four datasets considered, while achieving the best overall performance. Furthermore, our method shows the best performance on all three metrics with both models, which highlights its effectiveness in detecting OOD data.

## D.3 Temporal model ensemble

Following Pruthi et al. [37], we measure the label error detection performance by using the temporal model ensemble. Specifically, we average the label noisiness scores from 4 checkpoints that are uniformly sampled throughout training. Table 12 shows that this technique improves the performance of all methods, with our approach still exhibiting the best performance. These results confirm the effectiveness of temporal ensembles when more computation and storage are available.

## D.4 Compatibility term design

In Equation (2), we propose a compatibility term that measures a dot-product between prediction probability vectors. We denote this term as *Dot*. In Table 13, we evaluate other types of compatibility terms. Specifically, we consider a similarity kernel without the compatibility term, denoted as *None*, and also consider a distance-based compatibility term  $c(\mathbf{p}_i, \mathbf{p}_j) = \|\mathbf{p}_i - \mathbf{p}_j\|_2$ , referred to as *Distance*. Table 13 shows that our relation graph outperforms the best baseline regardless of the kernel design, especially on the TNR95 metric by a large margin. The results show that the dot-product compatibility term performs comparably to the distance-based compatibility term, slightly outperforming it on the AP metric. By comparing *None* to others, we verify the effectiveness of the compatibility term.



Table 13: Comparison of the compatibility terms. *Baseline* denotes the best baseline performance.

Task	Metric	Baseline	None	Distance	Dot
Label Error (8%)	AP	0.484	0.471	0.506	0.526
	TNR95	0.521	0.671	0.708	0.695
OOD (ALL)	AP	0.854	0.855	0.857	0.874
	TNR95	0.590	0.630	0.637	0.636

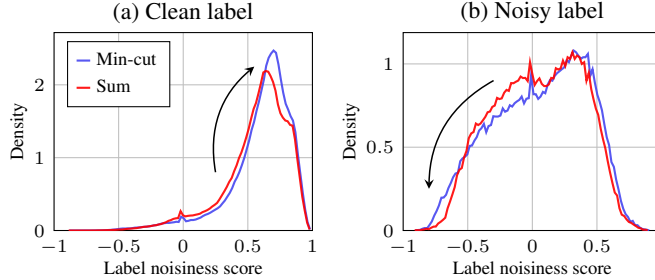


Figure 12: Normalized histogram of label noisiness scores by simple edge aggregation (*Sum*) and min-cut. Arrows indicate changes in the density functions.

## D.5 Analysis on edge aggregation

In this section, we analyze the difference between label noisiness scores calculated using min-cut and simple edge sum. Specifically, in Figure 12, we draw histograms of label noisiness scores for two groups of data: one with clean labels and the other with label errors. The figure shows that the score calculated through min-cut shifts the density in a positive direction for data with clean labels and in a negative direction for data with label errors, compared to the simple edge sum. This indicates that the label noisiness score calculated with min-cut better separates label errors from the clean data. In this case, TNR95 increases from 0.628 to 0.647 by applying the min-cut.

## E Additional qualitative results

### E.1 Relation map

In Figure 13, we present additional relation maps on ImageNet. We draw the relation maps with MAE-Large and ResNet-50. Note, MAE-Large utilizes masked auto-encoding pre-training whereas ResNet-50 is trained from scratch. From the figure, we observe that the models exhibit similar distributions of positive and negative relations for each data point. However, the ResNet model tends to have an overall larger variance of relation, indicating that the pre-training process reduces the relation variance and is helpful in forming relationships between data.

### E.2 Qualitative results

We present problematic data samples detected by our algorithm. Specifically, Figure 14 shows ImageNet samples with label errors and their most conflicting data samples having negative relation values. Figure 15 shows the outlier data detected by our algorithm on the ImageNet validation set, indicating the existence of inappropriate data for evaluation. We also present results on SST2 [53], a binary text sentiment classification dataset, in Tables 18 and 19.

Table 14: Label error detection performance on a range of label error ratios (MAE-Large, ImageNet).

Method \ Ratio	AUROC				AP				TNR95			
	4%	8%	12%	15%	4%	8%	12%	15%	4%	8%	12%	15%
Entropy	0.790	0.796	0.801	0.803	0.153	0.246	0.334	0.389	0.115	0.137	0.162	0.181
Least-conf.	0.829	0.836	0.843	0.847	0.176	0.282	0.380	0.443	0.223	0.270	0.327	0.368
CWE	0.845	0.845	0.845	0.844	0.301	0.397	0.471	0.506	0.135	0.160	0.188	0.212
Loss	0.864	0.864	0.865	0.865	0.328	0.424	0.497	0.530	0.229	0.276	0.334	0.374
TracIn	0.889	0.888	0.887	0.885	0.303	0.415	0.498	0.536	0.503	0.521	0.556	0.582
Margin	0.876	0.875	0.876	0.876	0.403	0.484	0.544	0.568	0.346	0.392	0.457	0.505
Relation (Ours)	<b>0.917</b>	<b>0.914</b>	<b>0.910</b>	<b>0.904</b>	<b>0.437</b>	<b>0.526</b>	<b>0.590</b>	<b>0.611</b>	<b>0.695</b>	<b>0.695</b>	<b>0.687</b>	<b>0.683</b>

Table 15: Label error detection performance according to the number of data (MAE-Large, ImageNet, 8% label noise).

Method \ #data	AUROC					AP					TNR95				
	12k	25k	100k	400k	1.2M	12k	25k	100k	400k	1.2M	12k	25k	100k	400k	1.2M
Entropy	0.806	0.801	0.797	0.796	0.796	0.250	0.248	0.248	0.246	0.246	0.161	0.161	0.144	0.137	0.137
Least-conf.	0.844	0.841	0.836	0.837	0.836	0.289	0.287	0.284	0.283	0.282	0.255	0.269	0.275	0.277	0.27
CWE	0.852	0.850	0.845	0.845	0.845	0.390	0.394	0.392	0.394	0.397	0.169	0.184	0.177	0.163	0.16
Loss	0.869	0.868	0.863	0.864	0.864	0.415	0.421	0.418	0.421	0.424	0.260	0.298	0.282	0.284	0.276
TracIn	0.891	0.890	0.886	0.887	0.888	0.408	0.415	0.409	0.412	0.415	0.554	0.541	0.523	0.524	0.521
Margin	0.878	0.878	0.874	0.875	0.875	0.481	0.486	0.477	0.481	0.484	0.376	0.424	0.412	0.398	0.392
Relation (Ours)	<b>0.910</b>	<b>0.912</b>	<b>0.912</b>	<b>0.914</b>	<b>0.914</b>	<b>0.502</b>	<b>0.515</b>	<b>0.518</b>	<b>0.523</b>	<b>0.526</b>	<b>0.679</b>	<b>0.692</b>	<b>0.692</b>	<b>0.696</b>	<b>0.695</b>

Table 16: OOD detection performance on ImageNet with MAE-Large.

Method \ OOD dataset	AUROC					AP					TNR95				
	ALL	Places	SUN	iNat.	Textures	ALL	Places	SUN	iNat.	Textures	ALL	Places	SUN	iNat.	Textures
MSP	0.857	0.835	0.833	0.907	0.853	0.818	0.543	0.567	0.699	0.514	0.428	0.386	0.315	0.651	0.376
Max Logit	0.824	0.787	0.790	0.881	0.850	0.808	0.531	0.557	0.700	0.570	0.138	0.106	0.088	0.320	0.216
Mahalanobis	0.875	0.819	0.841	0.948	0.904	0.815	0.471	0.489	0.722	0.532	0.521	0.407	0.474	<b>0.847</b>	0.599
Energy	0.776	0.728	0.738	0.829	0.837	0.757	0.446	0.467	0.592	0.553	0.074	0.060	0.050	0.127	0.154
ReAct	0.896	0.862	0.872	0.944	0.909	0.857	0.586	0.611	0.746	0.634	0.550	0.449	0.463	0.793	0.547
KL-Matching	0.877	0.848	0.857	0.928	0.874	0.818	0.500	0.526	0.708	0.455	0.548	0.473	0.490	0.738	0.511
Iterative sampling	0.444	0.377	0.409	0.457	0.600	0.382	0.127	0.134	0.145	0.173	0.073	0.051	0.061	0.125	0.131
Local outlier factor	0.556	0.492	0.521	0.594	0.665	0.431	0.151	0.160	0.183	0.173	0.156	0.113	0.136	0.283	0.240
KNN	0.901	0.861	0.884	0.946	<b>0.922</b>	0.854	0.550	0.593	0.736	<b>0.648</b>	0.590	0.487	0.560	0.808	0.626
Relation (Ours)	<b>0.911</b>	<b>0.883</b>	<b>0.894</b>	<b>0.951</b>	<b>0.921</b>	<b>0.874</b>	<b>0.618</b>	<b>0.653</b>	<b>0.782</b>	0.642	<b>0.636</b>	<b>0.547</b>	<b>0.587</b>	0.810	<b>0.641</b>

Table 17: OOD detection performance on ImageNet with ResNet-50.

Method \ OOD dataset	AUROC					AP					TNR95				
	ALL	Places	SUN	iNat.	Textures	ALL	Places	SUN	iNat.	Textures	ALL	Places	SUN	iNat.	Textures
MSP	0.847	0.829	0.836	0.896	0.814	0.782	0.469	0.501	0.641	0.369	0.496	0.471	0.461	0.634	0.352
Max Logit	0.844	0.827	0.833	0.894	0.807	0.767	0.445	0.468	0.605	0.331	0.482	0.463	0.448	0.622	0.340
Mahalanobis	0.693	0.644	0.628	0.735	0.823	0.567	0.228	0.209	0.270	0.375	0.265	0.194	0.229	0.436	0.371
Energy	0.836	0.821	0.825	0.883	0.798	0.721	0.398	0.407	0.503	0.269	0.481	<b>0.463</b>	0.448	0.622	0.340
ReAct	0.624	0.610	0.623	0.626	0.650	0.446	0.185	0.189	0.188	0.128	0.361	0.302	0.370	0.429	0.371
KL-Matching	0.838	0.811	0.822	0.894	0.817	0.784	0.439	0.463	0.693	0.404	0.307	0.268	0.224	0.477	0.291
Iterative sampling	0.624	0.530	0.578	0.659	0.809	0.516	0.167	0.190	0.229	0.380	0.150	0.112	0.134	0.223	0.269
Local outlier factor	0.648	0.556	0.602	0.686	0.824	0.533	0.175	0.199	0.244	0.397	0.186	0.143	0.167	0.272	0.315
KNN	0.822	0.743	0.783	0.884	<b>0.922</b>	0.764	0.335	0.384	0.546	<b>0.774</b>	0.380	0.295	0.372	0.587	<b>0.496</b>
Relation (Ours)	<b>0.870</b>	<b>0.830</b>	<b>0.853</b>	<b>0.922</b>	0.879	<b>0.818</b>	<b>0.493</b>	<b>0.543</b>	<b>0.708</b>	0.513	<b>0.515</b>	0.429	<b>0.498</b>	<b>0.691</b>	0.465

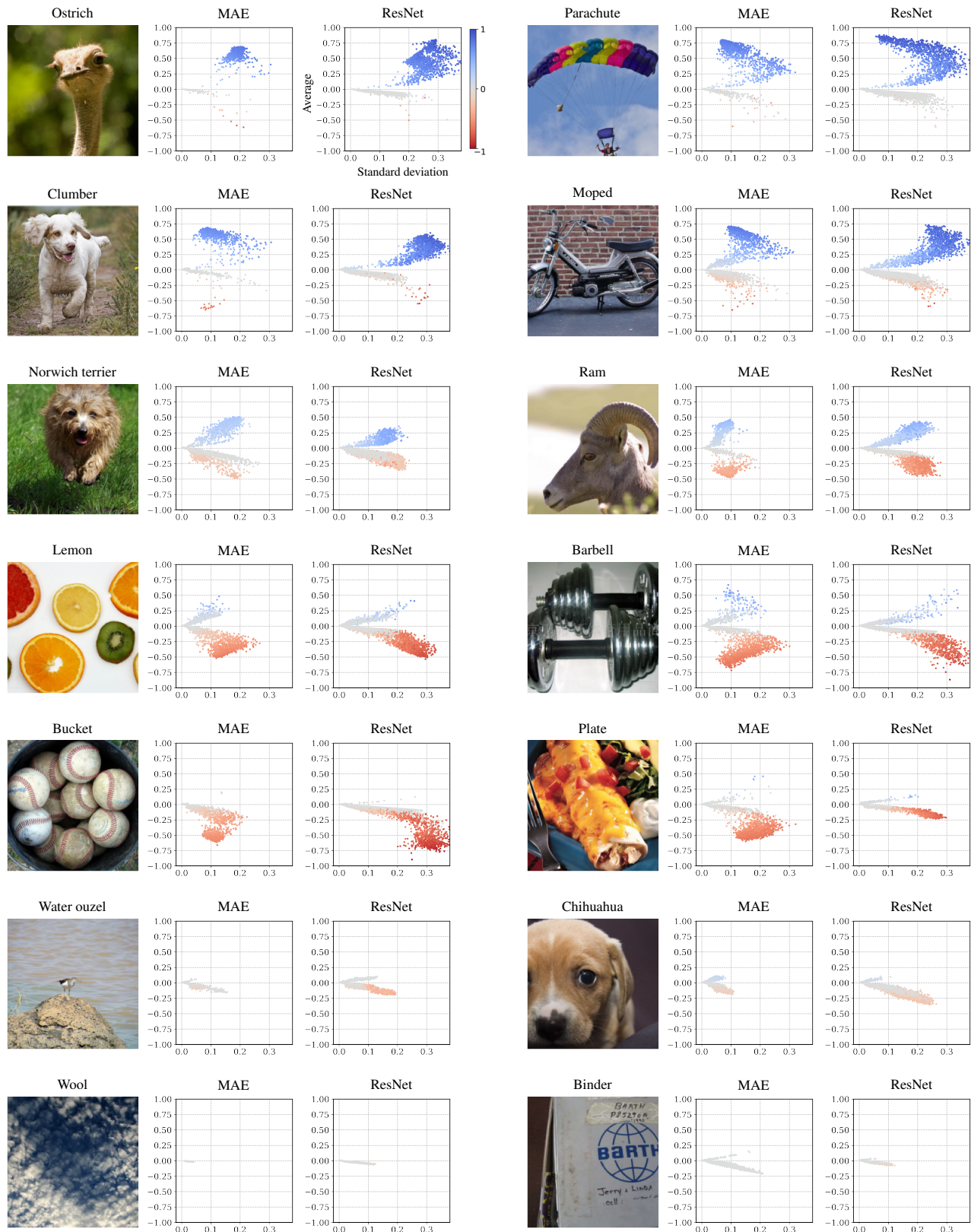


Figure 13: Data relation maps on ImageNet with MAE-Large and ResNet-50. We denote the assigned label above each image. The color represents the relation value at the last checkpoint. The x-axis is the standard deviation and the y-axis is the mean value of the relation values throughout training.

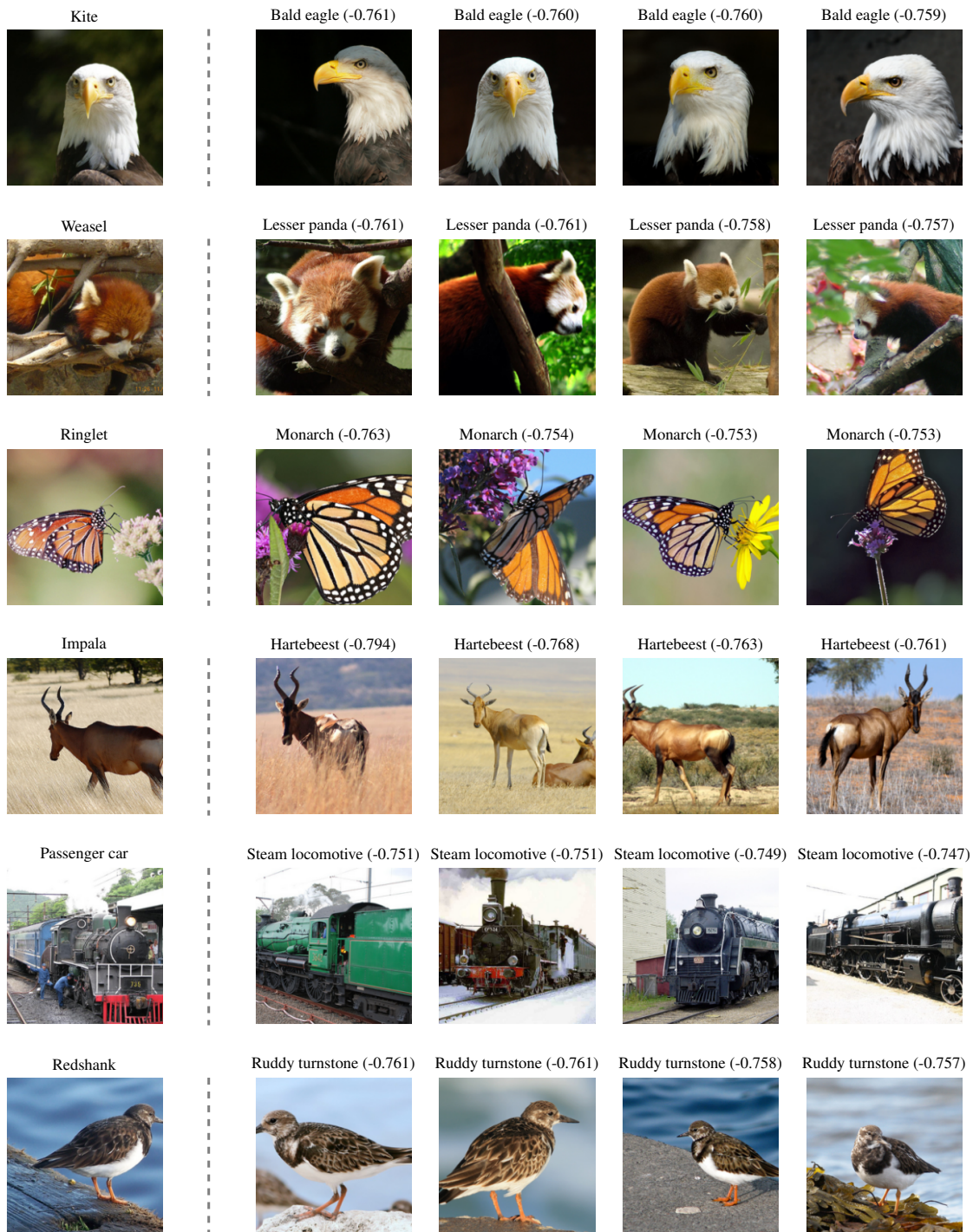


Figure 14: The first column shows the data samples detected by our label error detection algorithm using MAE-Large on ImageNet. We present the samples with the most conflicting relation next to the detected samples. We denote the assigned label and the corresponding relation value above the image.



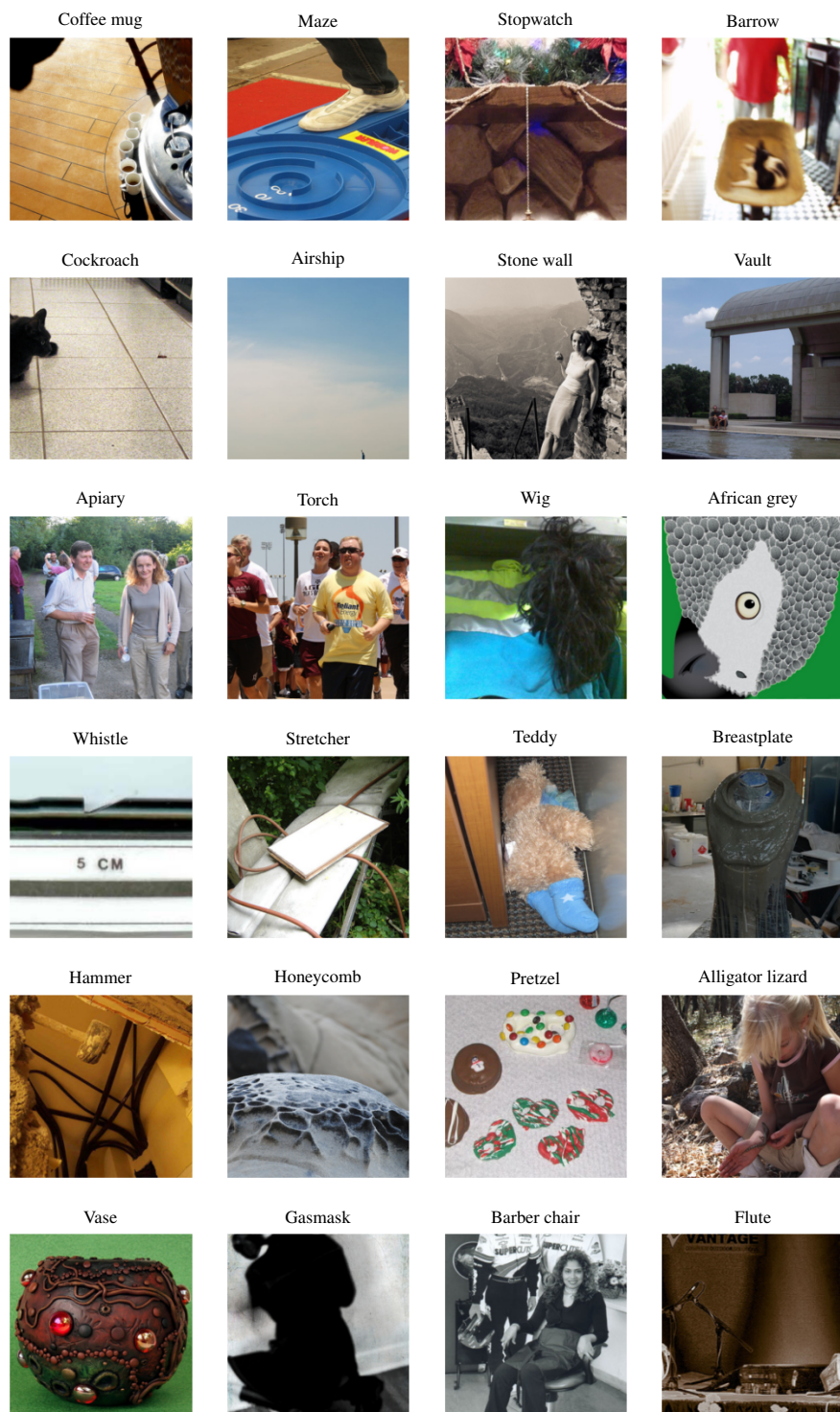


Figure 15: Data samples with the lowest outlier score by our method on the ImageNet validation set. We denote the assigned label above the image.

Table 18: Text samples with label errors detected by our algorithm on the SST2 dataset. Below the text with label error, we present two text samples with conflicting relations and denote the corresponding relation value in parenthesis.

Text	Label
a damn fine and a truly distinctive and a deeply pertinent film	Negative
- a breathtakingly assured and stylish work	Positive (-0.980)
- a winning and wildly fascinating work	Positive (-0.978)
fails to have a heart, mind or humor of its own	Positive
- failing to find a spark of its own	Negative (-0.970)
- this movie's lack of ideas	Negative (-0.958)
a ploddingly melodramatic structure	Positive
- plodding action sequences	Negative (-0.959)
- plodding picture	Negative (-0.957)
is somewhat problematic	Positive
- the more problematic aspects	Negative (-0.945)
- the problematic script	Negative (-0.930)
a bittersweet contemporary comedy	Negative
- bittersweet film	Positive (-0.940)
- of bittersweet camaraderie and history	Positive (-0.921)

Table 19: Outlier text samples detected by our algorithm on the SST2 dataset.

Text	Label
the battle	Negative
give a backbone to the company	Positive
leather pants	Positive
the israeli/palestinian conflict as	Negative
the story relevant in the first place	Positive
from a television monitor	Positive
loud, bang-the-drum	Positive
a movie instead of an endless trailer	Negative
a doctor's office, emergency room, hospital bed or insurance company office	Positive
this is more appetizing than a side dish of asparagus	Negative