Sample Selection via Contrastive Fragmentation for Noisy Label Regression

Chris Dongjoo Kim[1,2]* **Sangwoo Moon**[1]* **Jihwan Moon**[1]
Dongyeon Woo[1], **Gunhee Kim**[1,2]
[1]Seoul National University, [2]LG AI Research
{cdjkim, sangwoo.moon, jihwan.moon, dongyeon.woo}@vision.snu.ac.kr
gunhee@snu.ac.kr

Abstract

As with many other problems, real-world regression is plagued by the presence of noisy labels, an inevitable issue that demands our attention. Fortunately, much real-world data often exhibits an intrinsic property of continuously ordered correlations between labels and features, where data points with similar labels are also represented with closely related features. In response, we propose a novel approach named ConFrag, where we collectively model the regression data by transforming them into disjoint yet contrasting fragmentation pairs. This enables the training of more distinctive representations, enhancing the ability to select clean samples. Our ConFrag framework leverages a mixture of neighboring fragments to discern noisy labels through neighborhood agreement among expert feature extractors. We extensively perform experiments on six newly curated benchmark datasets of diverse domains, including age prediction, price prediction, and music production year estimation. We also introduce a metric called Error Residual Ratio (ERR) to better account for varying degrees of label noise. Our approach consistently outperforms fourteen state-of-the-art baselines, being robust against symmetric and random Gaussian label noise.[2].

1 Introduction

Regression is an important task in many disciplines such as finance [Zhang et al., 2017b, Wu et al., 2020b], medicine [de Vente et al., 2021, Tanaka et al., 2022], economics [Zhang et al., 2022], physics [Sia et al., 2020, Doi et al., 2022], geography [Liu et al., 2023] and more. However, real-world regression labels are prone to being corrupted with noise, making it an inevitable problem to overcome in practical applications. In previous research, noisy label regression has been studied much in age estimation with noise incurred from Web data crawling [Rothe et al., 2018, Yiming et al., 2021]. Beyond that, the issues of continuous label errors have also been reported in the tasks of object detection [Su et al., 2012, Ma et al., 2022] and pose estimation [Geng and Xia, 2014] as well as measurements in hardware systems [Zhou et al., 2012, Zang et al., 2019].

The vast amount of noisy label learning research has focused more on classification than regression. Some notable approaches include regularization [Wang et al., 2019, Zhang and Sabuncu, 2018], data re-weighting [Ren et al., 2018, Shen and Sanghavi, 2019], training procedures [Jiang et al., 2018], transition matrices [Yao et al., 2020, Xia et al., 2020], contrastive learning [Zhang et al., 2021a, Li et al., 2022b], refurbishing [Song et al., 2019] and sample selection [Lee et al., 2018, Ostyakov et al., 2018]. Particularly, sample selection can be further divided into exploring the memorability of neural

[*] These authors contributed equally to this work.
[2] The code is available at https://github.com/cdjkim/ConFrag

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

networks [Arpit et al., 2017, Zhang et al., 2017a] and delineating samples via the loss magnitude [Wei et al., 2020].

To the best of our knowledge, there have been three works that address the noisy label problem for regression with deep learning. Castells et al. [2020] propose a weighted loss correction method based on the small loss assumption. Garg and Manwani [2020] propose an ordinal regression-based loss correction via noise transition matrix estimation. However, they assume that accurate noise rates are known in prior [Patrini et al., 2017], which are hard to attain in practice. Yao et al. [2022] extend MixUp [Zhang et al., 2018] for regression to interpolate the proximal samples in the label space to improve generalization and robustness. Thanks to its regularizing effect, it can aid the noisy label issue.

In this work, we explore the regression problem with noisy labels, surpassing the scope of previous studies both empirically and methodologically. For evaluation, we make three notable contributions. Firstly, recognizing the absence of a standardized benchmark dataset for this task, we take the initiative to curate six balanced real-world datasets. These datasets span diverse domains, encompassing age estimation [Niu et al., 2016, Yiming et al., 2021], music production year estimation [Bertin-Mahieux et al., 2011], and clothing price prediction [Kimura et al., 2021]. Secondly, we conduct a comprehensive empirical benchmark exercise, evaluating the performance of fourteen baselines, which are carefully selected from various branches of noisy label research extendable to regression tasks. Lastly, we introduce a performance measure called Error Residual Ratio (ERR), which accounts for the unique property of regression, where labels exhibit varying degrees of noise severity.

Methodologically, we introduce the ConFrag (Contrastive Fragmentation) framework as a novel approach to address label noise in regression. It is rooted in one fundamental characteristic of regression: the continuous and ordered correlation between the label and feature space. In other words, data points similar in the feature space are likely to have similar labels. The framework begins by partitioning the dataset into smaller segments, referred to as fragments, and pairs the most distant fragments in the label space to form what we call *contrastive fragment pairs*. Training an expert network on these contrastive fragment pairs aids in generalization due to the distinctive feature matching and conversion of closed-set noise into open-set noise, which is less detrimental for learning. Next, the framework incorporates neighboring relationships by aggregating and reordering the learned features to detect clean samples. This is accomplished through the design of Mixture [Jacobs et al., 1991] of neighboring fragments. Furthermore, we enhance our approach with neighborhood jittering regularization, which strengthens the selection process by improving the data coverage of each expert. This, in turn, leads to improved agreements among neighboring fragments and serves as an effective tool for mitigating overfitting. Finally, the contributions of this work can be summarized as follows.

1. We propose a novel method named ConFrag (Contrastive Fragmentation) for noisy labeled regression. It leverages the inherent orderly relationship within the label and feature space by employing contrastive fragment pairing and constructs a mixture model based on neighborhood agreement. This is further enhanced by our neighborhood jittering regularization.
2. We perform one of the most thorough empirical investigations into noisy labeled regression up to date. We assemble six well-balanced benchmarks using datasets of AFAD [Niu et al., 2016], IMDB-Clean [Yiming et al., 2021], IMDB-WIKI [Rothe et al., 2018], UTK-Face [Zhifei et al., 2017], SHIFT15M [Kimura et al., 2021], and MSD [Bertin-Mahieux et al., 2011], on which we evaluate fourteen baselines. We design a metric termed ERR (Error Residual Ratio), which accounts for the degree of noise severity within the labels, offering a more comprehensive assessment. Our experiments affirm the superiority of ConFrag over state-of-the-art noisy label learning baselines.

2 ConFrag: Contrastive Fragmentation

In the noisy label regression problem, we are presented with a dataset denoted as $\mathcal{D} = \{\mathcal{X}, Y\}$; in each sample (x, y), $x \in \mathbb{R}^d$ is an input, and $y \in \mathbb{R}$ is the observed label, which can be possibly noisy. We use y^{gt} to denote the groundtruth label. The objective of ConFrag is to sample a *clean* subset of the data as $\mathcal{S} \subset \mathcal{D}$. By training on $\mathcal{S}$, we aim to enhance the performance of the regression model.

An overview of our ConFrag framework is shown in Fig. 3(a). The framework has the following steps. We divide the dataset into what we refer to as *contrastive fragment pairs* (§ 2.1), which collectively

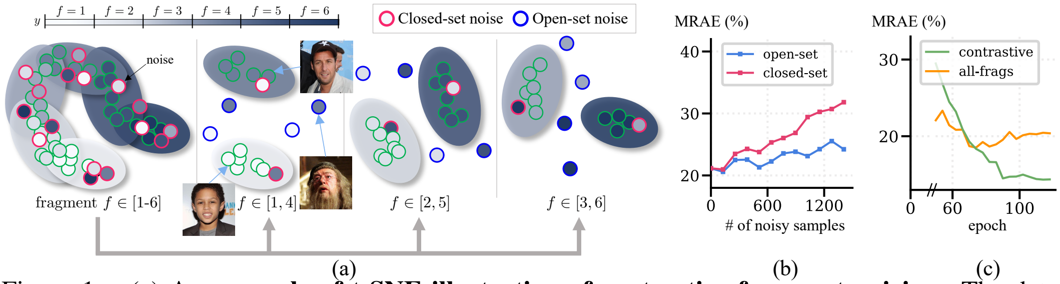
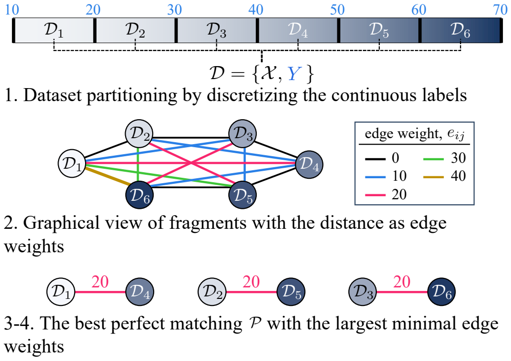


Figure 1: (a) **An example of t-SNE illustration of contrastive fragment pairing**. The data with label noise are grouped into six fragments ($f \in$ [1-6]) and formed into three contrastive pairs ($f \in [1, 4], [2, 5], [3, 6]$). Contrastive fragment pairing transforms some of closed-set noise (whose ground truth is within the target label set) into open-set noise (whose ground truth is not within the label set). For example, in the [1,4] figure, label noise whose ground truth fragment is either 1 or 4 is closed-set noise, and the others are open-set noise. The t-SNE illustration shows that learned features of open-set noises tend to reside outside the feature clusters of the clean samples. (b) The open-set noise is *less harmful* with much lower errors (MRAE) in the downstream regression. (c) The contrastive pairing ($[1, 4], [2, 5], [3, 6]$) is more effective than using all-fragments together ([1-6]), resulting in much lower MRAE scores. All experiments are based on IMDB-Clean-B with more details in Appendix G.4–G.5.

enhance the training of the feature extractors (§ 2.2). We then select clean samples $\mathcal{S}$ from dataset $\mathcal{D}$ based on neighborhood agreements, utilizing a fragment-based mixture model (§ 2.3). A regression model is trained on the clean samples $\mathcal{S}$. We also propose neighborhood jittering as a regularizer for further improved training (§ 2.4). ConFrag is noise rate-agnostic unlike prior methods as it operates without knowing a pre-defined noise rate.

2.1 Contrastive Fragment Pairing

In order to sample a *clean* data subset $\mathcal{S} \subset \mathcal{D}$, we need to learn a robust feature that can distinguish clean samples from noisy ones. As one theoretical result in [Zhang et al., 2023], the cross-entropy loss used in classification is better for learning high-entropy feature representation than the mean squared loss in regression (see Appendix D.1 for details). Based on this, we start by discretizing the label space into F continuous fragments, transforming the original regression problem into the multi-class classification one. This transformation harnesses an inherent property of regression: data points with similar labels are also represented with closely related features, as acknowledged in prior studies [Gong et al., 2022, Yang et al., 2022b, Yao et al., 2022].



1. Dataset partitioning by discretizing the continuous labels

2. Graphical view of fragments with the distance as edge weights

3-4. The best perfect matching $\mathcal{P}$ with the largest minimal edge weights

Figure 2: The contrastive fragment pairing algorithm.

However, instead of training a single feature extractor on the multi-class classification with F classes, we construct $F/2$ *maximally contrasting fragment pairs* and train a smaller expert feature extractor for each pair. The procedure of contrastive fragment pairing is detailed below with an illustration in Fig. 2:

1. Divide the range of continuous labels Y into F even number of equal-length fragments. This allows to divide the dataset $\mathcal{D}$ into F disjoint subsets: $\mathcal{D} = \{\mathcal{D}_1, ..., \mathcal{D}_F\}$, where each $\mathcal{D}_i$ contains the data samples whose y values are in the i-th fragment label range.
2. Construct a complete graph $g = \{\mathcal{D}, E\}$, where each vertex is a fragment $\mathcal{D}_i$, and each edge weight e_{ij} is the distance in the label space between the closest samples of the fragments $(\mathcal{D}_i, \mathcal{D}_j)$.
3. Compute all possible *perfect matchings* [Monfared and Mallik, 2016, Gibbons, 1985], where every vertex of a graph is incident to exactly one edge in the graph.

4. Find the perfect matching with the largest minimal edge weight: $\mathcal{P} = \arg\max_{\bar{g}\in\mathcal{G}}\Big(\min \nu(\bar{g})\Big)$, where each $\bar{g}$ is a perfect matching (graph), and $\nu(\bar{g})$ is the set of edge weights in $\bar{g}$. Finally, $\mathcal{P} = \{(\mathcal{D}_i, \mathcal{D}_j), \ldots, (\mathcal{D}_k, \mathcal{D}_l)\}$ constitutes the *maximally contrasting pairs* of fragments.

Motivation behind contrastive fragment pairing. Formulating the multi-class classification problem into $F/2$ binary classification problems via contrastive fragment pairing has the following advantages. Firstly, since the distance between fragments in each contrastive fragment pair is large, the feature extractor trained on each contrastive pair can generalize better [Shawe-Taylor and Cristianini, 1998, Grønlund et al., 2019, 2020]. Fig. 1(c) shows the generalization abilities of the expert feature extractors trained on contrastive fragment pairs compared to the single feature extractor trained on all fragments. When using a single feature extractor on all fragments (all-frags), the samples selected by the feature extractor tend to become more noisy as the feature extractor overfits, causing the regressor to perform worse over time. On the other hand, when using multiple feature extractors trained on contrastive pairs, the performance of the regression model consistently improves, indicating that the learned features are more robust and the selected samples are cleaner. The large distance between fragments also explains why contrastive fragment pairing is superior to other fragment pairings, as shown in § 4.3. The analysis of the prediction depth [Baldock et al., 2021] in Appendix D.3 supports the claim, as it shows that the binary classification on contrastive fragment pairs results in lower prediction depth, leading to better generalization.

Secondly, the contrastive fragment pairing transforms some of *closed-set* label noise (whose ground truth is within the label set) into *open-set* label noise (whose ground truth is not within the label set), as shown in Fig. 1(a). Previous works [Wei et al., 2021, Wan et al., 2024] observe that the open-set noise is less harmful than the closed-set noise and may even benefit generalization and robustness against inherent noisy labels. Indeed, in our experiments, we found similar observations where injecting open-set label noise is less harmful than closed-set one, as shown in Fig. 1(b).

The t-SNE visualization in Fig. 1(a) also supports this observation. Let f and f^{gt} be fragments that the observed label y and the groundtruth label y^{gt} respectively belong to. Prior to contrastive fragment pairing, all of the noisy labeled data ($f \neq f^{\text{gt}}$) are closed-set noise as their ground truth fragment ids are within the label set ($f^{\text{gt}} \in$ [1-6]) and their features are located in the feature spaces of incorrect classes within the group. After contrastive fragment pairing, much of these noisy labeled data is transformed to open-set noise ($f^{\text{gt}} \notin [1, 4]$ while $f \in [1, 4]$ in case of fragment pair $[1, 4]$), and their learned features tend to reside outside the feature clusters of the clean samples, thus mitigating the adverse effects of the noise.

2.2 Training Feature Extractors for Contrastive Pairs

Once we obtain the contrastive fragment pairs $\mathcal{P}$, we train $F/2$ number of expert feature extractors on binary classification $p(y|x; \theta_{i,j})$ with its respective contrastive pair $(\mathcal{D}_i, \mathcal{D}_j) \in \mathcal{P}$, where $\theta_{i,j}$ denotes the parameter of an expert. That is, it is trained to predict whether a data x is in $\mathcal{D}_i$ or $\mathcal{D}_j$. Later, the feature extractors play a crucial role in determining whether a sample (x, y) is clean.

2.3 Mixture of Neighboring Fragments

With the learned expert feature extractors, the next step is to perform sample selection. Given a sample (x, y), let f be a fragment close to y and f^+ be its contrasting pair. Intuitively, we consider a sample clean if the expert trained on $(\mathcal{D}_f, \mathcal{D}_{f+})$ strongly predicts that x belongs to a fragment f. However, since the expert feature extractor is a binary classifier only trained using a contrasting pair of fragments, we utilize all experts' opinions to obtain a more robust prediction. Specifically, we deem a sample as clean if the experts exhibit a consensus response (**Neighborhood Agreement**) for fragments close to y (**Fragment Prior**).

Based on this intuition, we formulate Mixture of Experts (MoE) [Jacobs et al., 1991] model, where the sampling probability of a datapoint (x, y) is defined as

$$p(s|x, y, \mathcal{D}_{1\ldots F}; \Theta) = \sum_{f}^{F} \rho_f(y)\alpha_f(x; \mathcal{D}_{1\ldots F}, \Theta), \tag{1}$$

where Θ denotes parameters of all feature extractors, ρ_f is the *fragment prior* (mixture weight), and α_f is the *neighborhood agreement* (a binary vote of whether x belongs to the fragment f). Based on

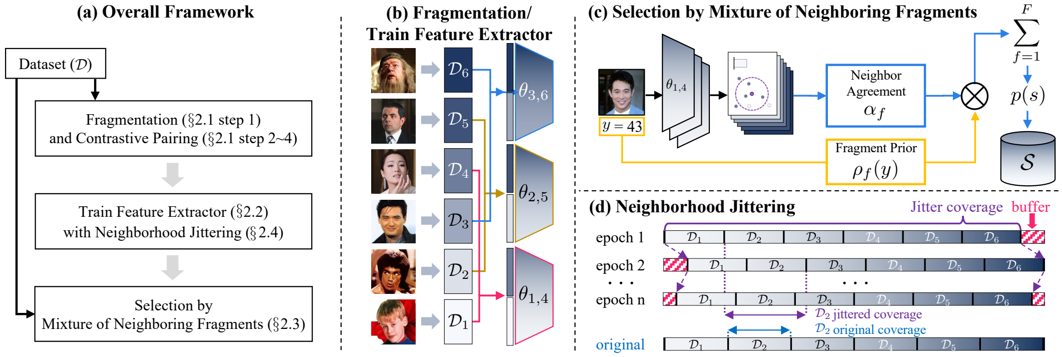

(a) Overall Framework
Dataset ($\mathcal{D}$)
Fragmentation (§2.1 step 1) and Contrastive Pairing (§2.1 step 2~4)
Train Feature Extractor (§2.2) with Neighborhood Jittering (§ 2.4)
Selection by Mixture of Neighboring Fragments (§2.3)
(b) Fragmentation/ Train Feature Extractor
$\mathcal{D}_6$
$\mathcal{D}_5$
$\mathcal{D}_4$
$\mathcal{D}_3$
$\mathcal{D}_2$
$\mathcal{D}_1$
$\theta_{3,6}$
$\theta_{2,5}$
$\theta_{1,4}$
(c) Selection by Mixture of Neighboring Fragments
$y = 43$
$\theta_{1,4}$
Neighbor Agreement α_f
Fragment Prior $\rho_f(y)$
$\sum_{f=1}^{F}$
$p(s)$
$\mathcal{S}$
(d) Neighborhood Jittering
Jitter coverage
buffer
epoch 1
epoch 2
epoch n
original
$\mathcal{D}_1$ $\mathcal{D}_2$ $\mathcal{D}_3$ $\mathcal{D}_4$ $\mathcal{D}_5$ $\mathcal{D}_6$
$\mathcal{D}_2$ jittered coverage
$\mathcal{D}_2$ original coverage

Figure 3: **Contrastive Fragmentation framework.** (a) The overall sequential process of our framework. (b) Shows the fragmentation of the continuous label space to obtain *contrasting fragment pairs* (§ 2.1) and train feature extractors on them. (c) Sample Selection by Mixture of Neighboring Fragments obtains the selection probability in both prediction and representation perspectives (§ 2.3). (d) Illustration of Neighborhood Jittering (§ 2.4).

the intuition above, $\rho_f(y)$ is large when the fragment f is close to y, and $\alpha_f(x) \in \{0, 1\}$ is 1 if x is likely to belong to the fragment f.

Fragment Prior. For a sample (x, y), we compute the prior $\rho_f(y)$ of a fragment f, using a softmax weighting of each fragment f with respect to its relative distance to y:

$$\rho_f(y) = \frac{\exp(g_f(y))}{\sum_{f'}^{F} \exp(g_{f'}(y))}, \tag{2}$$

where $g_f(y) = \text{range}(Y)/(|y - \bar{Y}_f|)$, $\text{range}(Y) = \max(Y) - \min(Y)$ is the label range, and $\bar{Y}_f$ is the mean label value of fragment f. Since $\text{range}(Y)$ is a constant for a given dataset, $g_f(y)$ rapidly decreases when the mean value of fragment f is far from y in the continuous label space. From the MoE perspective, the fragment prior can be regarded as soft gating that depends on y.

Neighborhood Agreement. Given a sample (x, y) and a fragment f, we need to determine whether x belongs to f. The simplest approach is to use the expert trained using $(\mathcal{D}_f, \mathcal{D}_{f^+})$ to classify whether x belongs to f or f^+, where f^+ is the contrasting fragment of f. Based on the classification output $h(x; \theta_{f,f^+}) \in \{f, f^+\}$, we define self-agreement as:

$$\alpha_f^{\text{self}} = [h(x; \theta_{f,f^+}) = f] \tag{3}$$

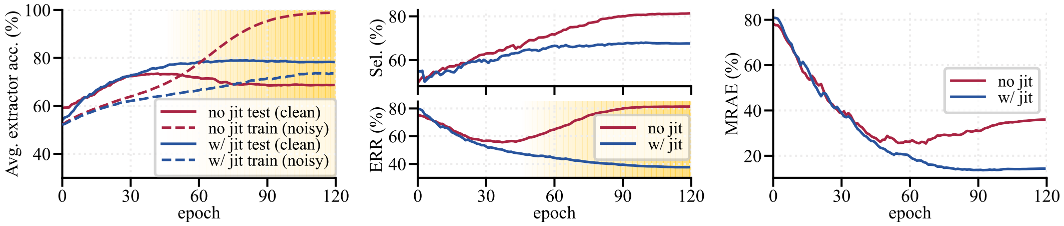
where $[A]$ is the Iverson bracket outputting 1 if A is true, and 0 otherwise. Since training with noisy labels often results in suboptimal calibration [Bae et al., 2022, Zong et al., 2024], we use discrete classification output for α_f^{self} rather than continuous probabilistic one.

Since the expert θ_{f,f^+} is only trained to discriminate between f and its contrasting fragment f^+, it is better to utilize other experts to obtain a more robust prediction. For example, consider contrastive fragment pairs $\{(1, 4), (2, 5), (3, 6)\}$ as in Fig. 2. If x is more likely to belong to fragment 2 than 5, then it should be more likely to belong to 1 than 4 and 3 than 6. Thus, we consider agreement of neighboring fragments f_L (left) and f_R (right) to obtain neighborhood agreement $\alpha_f(x; \mathcal{D}_{1\ldots F}, \Theta)$:

$$\alpha_f(x; \mathcal{D}_{1\ldots F}, \Theta) = \alpha_f^{\text{self}} \cdot \alpha_f^{\text{ngb}}, \quad \text{where} \;\; \alpha_f^{\text{ngb}} = \left[\alpha_{f_L}^{\text{self}} \vee \alpha_{f_R}^{\text{self}}\right]. \tag{4}$$

Intuitively, α_f is 1 if the fragment f is more likely for x than f^+ ($\alpha_f^{\text{self}} = 1$) and either f's left or right fragment is more likely for x than its respective contrasting fragment ($\alpha_f^{\text{ngb}} = 1$).

In practice, we implement two variants of the agreements in Eq.(3–4) using the feature extractor's binary classifier and a K-nearest neighbor classifier on the learned feature space. These two classifiers respectively consider *predictive* and *representational* aspects of the expert feature extractor and effectively work as an ensemble, as shown in Appendix G.7. As a result, we compute two versions of sample probability in Eq.(1), and use the union of the sampled *clean* dataset $\mathcal{S}$ for training of the regression model. Algorithm 1 in Appendix summarizes the overall procedure.



(a) Feature extractor acc. (b) Selection rate / ERR (c) Regression performance

Figure 4: **Jittering analysis.** (a) When trained without jittering, feature extractors easily overfit the noisy training data (yellow-shaded region), while jittering-regularized feature extractors robustly learn from the noisy training data. (b) Overfitted feature extractors (yellow-shaded region) on noisy samples increase their likelihood, leading to a higher selection rate and ERR. It exhibits nearly twice higher ERR (a lower value is better). (c) Most importantly, jittering regularization improves performance in regression.

2.4 Neighborhood Jittering

A potential limitation of mixture models is that the individual expert feature extractor may not fully benefit from the full dataset as they model their own disjoint subsets [Dukler et al., 2023]. Our neighborhood jittering mitigates this limitation as a robust regularizer that expands the effective coverage of each contrastive fragment pair during learning. The process is visualized in Fig. 3(d).

We bound the ratio of the jittering buffer range within $[0, \frac{1}{2(F-1)}]$, where F is the fragment number. For every epoch, we shift the label coverage of each fragment by randomly sampling the value in this range. Jittering leads to a partially overlapping mixture model [Heller and Ghahramani, 2007b, Hinton, 2002] as some data belong to multiple, neighboring fragments and thus the effective coverage per each expert is expanded. Such regularization inhibits feature extractors from overfitting to potentially noisy samples and promotes learning of more robust features, even those that can be generalizable to overlapping parts of neighboring fragments.

Fig. 4(a) shows that with jittering, the feature extractor exhibits higher accuracy on the clean test data due to its regularization effect. In the sample selection stage (Fig. 4(b)), the feature extractor trained without jittering easily overfits the noise, resulting in over-selection and higher ERR (§ 4.2). In contrast, the jittered feature extractor achieves a relatively low selection rate with halved ERR, indicating that the noisier samples are filtered out. Better sample selection due to jittering subsequently leads to significantly better performance in regression (Fig. 4(c)). In Appendix G.9, we compare neighborhood jittering to other regularizations, demonstrating its efficacy.

3 Related Works

We review prior works on learning with noisy labels and defer a comprehensive survey to Appendix E. We organize them into those utilizing prediction, representation, and combination of the two.

Prediction-based Methods. This approach has been the focus of much existing research and covers a wide array of topics: (i) the small loss selection by exploring the pattern of memorization in neural networks [Han et al., 2018, Arazo et al., 2019], (ii) relying on the consistency of predictions to select or refurbish the samples [Liu et al., 2020, Huang et al., 2020], (iii) estimating the noise distribution [Patrini et al., 2017, Hendrycks et al., 2018], (iv) introducing auxiliary parameters or labels [Pleiss et al., 2020, Hu et al., 2020], (v) using unlabeled data with semi-supervised learning [Li et al., 2020a, Bai et al., 2021, Karim et al., 2022], and (vi) designing a noise-robust loss function [Menon et al., 2020, Wang et al., 2019].

Representation-based Methods. This approach has seen a recent surge in interest, including (i) clustering based selection [Mirzasoleiman et al., 2020, Wu et al., 2020a], (ii) feature eigendecomposition filtering [Kim et al., 2021], (iii) using neighbor information to sample and refurbish with clean validation [Li et al., 2022a, Gao et al., 2016], and (iv) generative models of features for sampling [Lee et al., 2019].

Combination. Some works have also studied the combination of representation and prediction spaces. Wang et al. [2022] formulate a penalized regression between the network features and the labels for selection, and Ma et al. [2018] use intrinsic dimensionality and consistent predictions to refurbish. Other important approaches include (i) regularization via MixUp [Zhang et al., 2018] along with its regression version [Yao et al., 2022], (ii) model-based methods that discourage large parameter shifts [Hu et al., 2020], and (iii) importance discrimination of parameter updates [Xia et al., 2021].

The majority of previous works have studied noisy labels for classification. Hence, a large portion of these works may not be directly applicable to regression tasks due to the restricted usage of class-wise information. In § 4, we empirically compare our method with some of these works that are expandable to the regression task with some or minor technical adaptation.

4 Experiments

We compare ConFrag with fourteen strong baselines adapted for noisy label regression. Due to the scarcity of benchmark datasets, we update existing datasets for the study of noisy labels.

4.1 Settings

Curation of Benchmark Datasets. We create six benchmark datasets for noisy labeled regression to encompass a sufficient quantity of balanced data, span multiple domains, and present a meaningful level of complexity. (i) *Age Prediction* from an image is a well-studied regression problem [Li et al., 2019, Shin et al., 2022, Lim et al., 2020]. To address this domain, we acquire four datasets of **AFAD** [Niu et al., 2016], **IMDB-Clean** [Yiming et al., 2021], **IMDB-WIKI** [Rothe et al., 2018], and **UTKFace** [Zhifei et al., 2017]. Notably, IMDB-WIKI contains real-world label noise stemming from the automatic web crawling process [Yiming et al., 2021]. We use a ResNet-50 backbone for all datasets. (ii) *Commodity Price Prediction* is a vital real-world task [Wen-Huang et al., 2021]. We opt for the **SHIFT15M** dataset [Kimura et al., 2021] due to the diversity and scale of this domain. This dataset is provided as the penultimate feature of the ImageNet pre-trained VGG-16 model. Consequently, we use a three-layer MLP architecture for all experiments [Papadopoulos et al., 2022, Kimura et al., 2021]. (iii) *Music Production Year Estimation* uses the tabular **MSD** dataset [Bertin-Mahieux et al., 2011]. This dataset is identified as one of the most intricate and challenging datasets, based on the test R2 score [Grinsztajn et al., 2022]. We adopt a tabular ResNet proposed by Gorishniy et al. [2021]. The suffix "-B" is appended to the dataset name (*e.g.*, AFAD-B) to indicate that it is a curated version of the original dataset. To focus on the noisy label problem, we take measures to balance the datasets as elaborated in Appendix F.1.

Experimental Design. For all datasets except for IMDB-WIKI-B which contains real-world label noise, we inject symmetric and Gaussian noise into the labels, as done in prior literature [Yao et al., 2022, Yi and Wu, 2019, Wei et al., 2020]. These types of noise can simulate a low-cost (human-free) controlled setting. Symmetric noise mimics randomness such as Web crawling or annotator errors, and Gaussian noise is often used for modeling the regression label noise. While Yao et al. [2022] inject a *fixed* 30% standard deviated Gaussian noise for *every label*, we make it more realistic by *randomizing* the standard deviation up to 30% or 50% of the domain's range. For our ConFrag experiments, we fix the fragment number (F) as four. See Appendix F.3 for further training details.

Baselines. There are many existing methods of noisy labeled learning for classification. We assess fourteen baselines from the three branches that are naturally adaptable to regression with minor or no updates. (i) Small loss selection: CNLCU-S,H [Xia et al., 2022], Sigua [Han et al., 2020], SPR [Wang et al., 2022], BMM [Arazo et al., 2019], DY-S [Arazo et al., 2019], SuperLoss [Castells et al., 2020]. (ii) Regularization: C-mixup [Yao et al., 2022], RDI [Hu et al., 2020], CDR [Xia et al., 2021], D2L [Ma et al., 2018]. (iii) Refurbishment: AUX [Hu et al., 2020], Selfie [Song et al., 2019], Co-Selfie [Song et al., 2019]. Appendix F.2 comprehensively details these baselines.

4.2 Evaluation Metrics

We mainly report the Mean Relative Absolute Error (MRAE) following prior works. The MRAE is computed as $(e/\rho) - 1$, where e is the model's Mean Absolute Error (MAE) performance under varying conditions (noise type, severity) and ρ is the noise-free model's MAE. We express MRAEs in percentage for better comprehensibility. The traditional MAE values are also reported in Appendix G.12. In addition, we report the Selection rate (a.k.a prevalence), which is a metric often seen

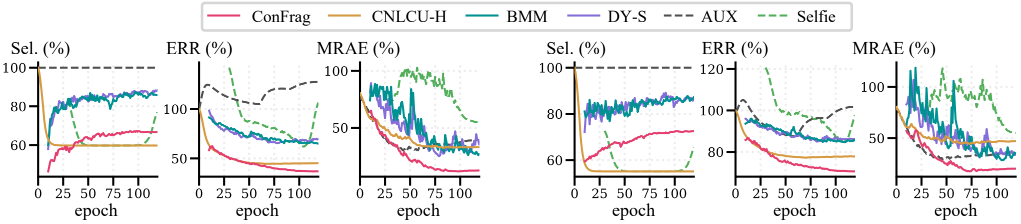
Table 1: **Comparison of Mean Relative Absolute Error (%)** over the noise-free trained model on the AFAD-B, IMDB-Clean-B, IMDB-WIKI-B, SHIFT15M-B, and MSD-B datasets. Lower is better. A negative value indicates it performs even better than the noise-free model. The results are the mean of three random seed experiments. The best and the second best methods are respectively marked in red and blue. CNLCU-S/H, Co-Selfie, and Co-ConFrag use dual networks to teach each other as done in Han et al. [2018]. SPR [Wang et al., 2022] fails to run for SHIFT15M-B due to excessive memory usage.

	AFAD-B						IMDB-Clean-B						IMDB-WIKI-B
	symmetric				Gaussian		symmetric				Gaussian		real noise
noise rate	20	40	60	80	30	50	20	40	60	80	30	50	-
Vanilla	9.37	20.27	30.65	43.09	28.77	39.03	16.18	32.05	53.13	76.35	26.89	50.28	0
CNLCU-S	10.98	20.44	32.44	41.99	30.60	40.66	51.40	66.62	82.83	85.65	83.39	82.10	21.54
CNLCU-H	4.63	16.32	36.01	44.71	35.68	43.64	6.84	31.16	63.08	82.65	46.53	65.24	-2.93
Sigua	5.96	21.09	43.33	49.71	42.52	46.19	9.82	46.17	77.59	85.62	60.97	77.42	1.96
SPR	9.74	18.85	30.43	43.25	28.50	39.69	14.47	32.44	54.88	79.37	25.67	51.05	-0.93
BMM	5.60	15.00	39.15	46.41	30.96	44.00	8.85	21.54	55.57	80.40	24.33	57.21	17.88
DY-S	6.87	15.56	32.24	45.72	24.40	43.41	10.42	21.90	49.94	78.16	24.70	44.56	-3.41
C-Mixup	2.74	14.80	27.17	41.95	24.28	36.91	8.82	27.74	50.87	76.79	21.92	47.04	-5.26
RDI	10.64	21.80	39.32	47.07	37.33	44.41	16.35	29.33	55.91	79.92	25.69	51.35	1.06
CDR	10.26	18.71	32.27	43.38	29.74	39.21	17.47	32.19	54.75	75.45	28.46	51.73	-0.39
D2L	9.43	20.75	31.25	44.50	28.86	40.10	16.94	33.85	55.54	76.28	29.30	52.44	-0.66
AUX	6.15	19.01	31.16	42.83	28.28	39.05	12.58	28.82	52.33	76.75	23.27	49.42	-3.67
Selfie	16.91	25.02	44.18	47.78	46.02	50.73	27.43	53.74	79.38	84.00	60.68	78.03	14.00
Co-Selfie	14.61	22.95	39.79	47.72	41.05	53.00	23.52	50.07	67.42	84.25	52.44	74.73	-0.44
Superloss	7.36	18.24	29.78	44.26	27.59	42.96	23.38	45.41	67.11	80.85	53.88	63.33	-3.58
ConFrag	2.74	8.16	15.91	34.42	17.49	27.31	5.08	12.64	27.26	61.24	15.70	33.36	-3.06
Co-ConFrag	0.54	7.25	16.65	33.93	17.43	28.26	1.50	9.45	28.44	61.36	14.87	35.88	-8.86

	SHIFT15M-B						MSD-B					
	symmetric				Gaussian		symmetric				Gaussian	
noise rate	20	40	60	80	30	50	20	40	60	80	30	50
Vanilla	9.11	17.96	27.02	36.34	6.54	15.16	8.23	18.43	31.67	45.85	6.96	15.74
CNLCU-S	12.98	19.42	24.31	34.47	15.33	20.90	0.13	6.04	21.52	46.01	4.75	12.51
CNLCU-H	6.26	12.84	20.04	36.03	8.88	15.65	0.27	4.98	10.32	29.83	5.11	9.22
Sigua	6.94	14.09	26.08	37.03	10.32	17.44	1.29	7.19	17.35	50.87	6.80	12.38
SPR	-	-	-	-	-	-	7.07	18.19	33.39	45.61	5.01	15.36
BMM	6.96	12.42	18.64	26.79	7.58	13.13	3.32	10.30	23.40	43.56	5.29	11.85
DY-S	7.11	11.94	18.85	29.04	6.90	13.50	3.39	8.06	18.65	35.24	4.77	9.83
C-Mixup	9.47	16.15	24.08	34.17	5.88	14.51	3.75	13.13	26.73	40.90	2.96	10.97
RDI	9.91	17.92	26.63	36.29	7.08	15.18	21.04	30.09	38.78	49.49	19.19	27.88
CDR	9.52	17.78	26.97	35.97	7.14	15.17	7.83	17.86	32.83	45.91	6.73	16.92
D2L	9.25	18.03	26.55	36.23	6.34	15.60	7.13	19.96	32.47	46.64	5.51	15.54
AUX	7.74	16.95	26.61	36.47	4.92	14.40	6.12	18.18	31.09	45.70	5.21	15.45
Selfie	4.84	10.22	22.28	38.15	5.51	11.58	1.43	8.40	20.24	45.87	14.37	24.13
Co-Selfie	11.53	16.43	32.08	39.32	13.45	22.33	-0.38	4.41	8.32	35.47	6.78	13.15
Superloss	5.44	12.26	23.23	35.24	5.60	13.28	-0.15	10.68	23.15	45.55	4.35	16.36
ConFrag	2.46	6.18	10.68	19.04	3.66	8.09	0.57	4.94	11.22	23.41	2.39	6.49
Co-ConFrag	0.85	5.52	10.80	18.83	3.03	8.70	-0.65	2.98	8.66	20.53	1.73	6.00

in noisy labeled classification to quantify the coverage of the total dataset, $|\mathcal{S}|/|\mathcal{D}|$ where $\mathcal{S}$ and $\mathcal{D}$ are the selected and total set, respectively.

Error Residual Ratio. To better assess selection and refurbishment approaches, we introduce a new metric termed Error Residual Ratio (ERR). Unlike classification, noisy labels in regression can show the diverse severity of the noise present in each label y (*i.e.*, various degrees of deviation from the ground truth y^{gt}). This cannot be addressed when using conventional metrics, which are primarily



(a) Symmetric 40%

(b) Gaussian 30

Figure 5: **Selection/ERR/MRAE comparison** between ConFrag and strong baselines of CNLCU-H, BMM, DY-S, AUX and Selfie on IMDB-Clean-B. We exclude the performance during the warm-up.

designed for classification and tend to treat all instances of noise as equally severe. Our proposed ERR considers the varying severity of noise and is defined as

$$\mathrm{ERR} = \frac{1/|\mathcal{C}| \sum_{c}^{|\mathcal{C}|} |y_c - y_c^{\mathrm{gt}}|}{1/|\mathcal{D}| \sum_{d}^{|\mathcal{D}|} |y_d - y_d^{\mathrm{gt}}|}, \tag{5}$$

where $\mathcal{C}$ is a set of cleaned (selected or refurbished) samples. The numerator is the average cleaned error that serves as an indicator of the precision of the cleaned data, while the denominator is the average dataset error that normalizes it for standardized assessment. The ERR, along with the selection rate and regression metrics (*e.g.*, MSE, MRAE), provides a deeper insight into the model performance. Ideally, a method with a high selection rate coupled with low ERR and regression error can be deemed as closer to the upper bound.

4.3 Results and Discussion

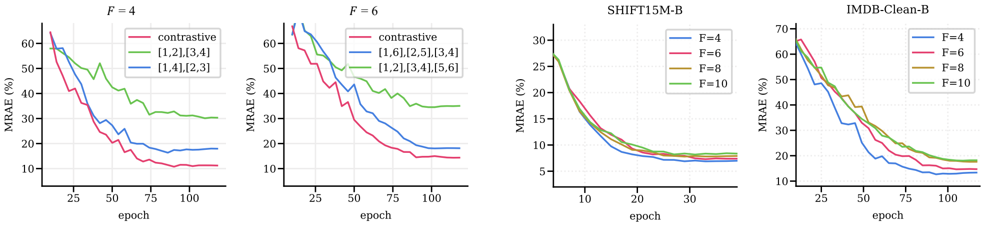
Overall performance. Table 1 compares the MRAE values to the noise-free trained model between ConFrag and the baselines. We evaluate six types of noise: four symmetric and two random Gaussian noises. ConFrag and Co-ConFrag achieve the strongest performance in all experiments compared to the fourteen baselines. Notably, Co-ConFrag mixes co-teaching during the regression learning phase by assuming that $\mathcal{S}$ still contains 25% noise. The results on UTKFace-B dataset can be found in Appendix G.1.

Selection/ERR/MRAE comparison. Fig. 5 compares ConFrag to five selection and refurbishment baselines of CNLCU-H, BMM, DY-S, AUX, Selfie on IMDB-Clean-B using the selection rate, ERR, and MRAE. Ideally, a model should attain a high selection rate and a low ERR. It is worth noting that the relative importance of ERR and selection rate may vary depending on the dataset and the task. ConFrag achieves the lowest ERR while maintaining above-average selection rates, resulting in the best MRAE. Appendix G.10 includes comparison results for all noise types with more baselines.

Fragment pairing. Fig. 6(a) compares contrastive pairing to alternative pairings using MRAE as a metric. The contrastive fragment pairing demonstrates superior performance to other pairing methods. Notably, the performance is poorest when both the average and minimum distance between fragments are smallest ($[1,2],[3,4]$ when $F=4$, $[1,2],[3,4],[5,6]$ when $F=6$). While the pairings of $[1,4],[2,3]$ and $[1,6],[2,5],[3,4]$ have the same average distance between fragments as the contrastive pairings, their minimum distances between fragments are smaller, resulting in poorer performances than contrastive pairings. This result shows the effectiveness of contrastive fragment pairing for selecting clean samples. See Appendix G.4 for more details.

Fragment number. ConFrag introduces a hyperparameter F, the number of fragments. While we simply set $F=4$ for all experiments, we conduct analysis on the effect of using different F, as shown in Fig. 6(b). On SHIFT15M-B dataset, the performance is relatively stable across different fragment numbers. On IMDB-Clean-B, a small declining trend in performance is observed as the number of fragments increases. This decrease is likely attributed to a finer division of the training data among feature extractors, ultimately leading to overfitting and reduced generalization capabilities. Appendix G.2 provides further analysis of the fragment number.

Ablation analysis on mixture of neighboring fragments. In Table 2, we conduct an ablation analysis of the Mixture of neighboring fragments (§ 2.3). When evaluating neighborhood agreement



(a) Fragment pairing analysis

(b) Fragment number analysis

Figure 6: **Analysis** with 40% symmetric noise. (a) Comparison between the proposed contrastive pairing and other pairings on IMDB-Clean-B. (b) Comparison between fragment numbers on SHIFT15M-B and IMDB-Clean-B.

Table 2: **Ablation of Mixture of Neighboring Fragments.** MRAE on the IMDB-Clean-B dataset (lower is better).

			symmetric	Gaussian	
α_f^{self}	α_f^{ngb}	$\mathcal{S}$	40	30	50
✓		$\mathcal{S}^p \cup \mathcal{S}^r$	16.97	17.53	38.18
	✓	$\mathcal{S}^p \cup \mathcal{S}^r$	22.22	22.77	46.36
✓	✓	$\mathcal{S}^p$	14.18	15.84	33.07
✓	✓	$\mathcal{S}^r$	14.06	16.94	39.85
✓	✓	$\mathcal{S}^p \cap \mathcal{S}^r$	13.08	16.18	34.23
✓	✓	$\mathcal{S}^p \cup \mathcal{S}^r$	12.64	15.70	33.36

Table 3: **Parameter size comparison**. regression: parameters for regression, noise: parameters to mitigate noisy labels, “others": SPR, CDR, D2L, C-Mixup, Sigua, Selfie, BMM, DY-S, Superloss.

	regression	noise	total
RDI	23.9M	47.8M	47.8M
CNLCU	47.8M	47.8M	47.8M
“others"	23.9M	23.9M	23.9M
ConFrag	23.9M	22.8M	46.7M

based solely on either the agreement of the current fragment (α_f^{self}) or the neighboring fragment's agreement (α_f^{ngb}), the ablation reveals that relying on the current fragment's agreement alone (α_f^{self}) exhibited relatively stronger performance. Nevertheless, this approach still fell short of achieving a satisfactory level compared to considering both agreements, as defined in Eq. 4.

Next, as we consider sample selection based on two variants of agreements, the *predictive* one utilizing the feature extractor's binary classifier and the *representational* one using K-nearest neighbors on the learned feature space (referred to as the selected sample sets $\mathcal{S}^p$ and $\mathcal{S}^r$ respectively), we conduct an ablation study on these selected sample sets. This involves evaluating the results when determining the final selected sample set ($\mathcal{S}$) either individually, at the intersection, or at the union of $\mathcal{S}^p$ and $\mathcal{S}^r$. Overall, in line with ConFrag, the union of sets ($\mathcal{S}^p \cup \mathcal{S}^r$) proves to be the most effective strategy.

Parameter size comparison. Table 3 compares the number of parameters of ConFrag and baselines on the ResNet-based age prediction datasets. A thorough description of the ConFrag architecture is in Appendix F.3. It is worth noting that each of the ConFrag's feature extractors for noise mitigation employs a much fewer number of parameters than the downstream regression task (*e.g.*, 48% in age prediction datasets). The total number of parameters of each method varies, as some share parameters for regression as well as noise mitigation while others, such as ConFrag, do not. Nevertheless, ConFrag uses fewer total parameters than CNLCU-H and RDI.

5 Conclusion

To address the problem of noisy labeled regression, we introduce the Contrastive Fragmentation framework (ConFrag). The framework partitions the label space and identifies the most contrasting pairs of fragments, thereby training a mixture of feature extractors over contrastive fragment pairs. This mixture is leveraged for clean selection based on neighborhood agreements. Extensive experiments on six curated datasets on three domains with different levels of symmetric and Gaussian noise demonstrate that our framework performs superior selection and ultimately leads to a better regression performance than fourteen state-of-the-art models. Given its foundation in the Mixture of Experts model, the parameter size of ConFrag linearly grows with an increase in the number of fragments. We acknowledge this as a potential avenue for future research.

Acknowledgement

We express our gratitude to the members of the Vision and Learning Lab at Seoul National University for their valuable feedback on the manuscript. In particular, we would like to acknowledge Jaekyeom Kim, Soochan Lee, Jinseo Jeong, Wonkwang Lee, and Sehun Lee. This work was supported by LG AI Research, Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2019-II191082, SW StarLab), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2022-II220156, Fundamental research on continual meta-learning for quality enhancement of casual videos and their 3D metaverse transformation), the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2023R1A2C2005573), and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)). Gunhee Kim is the corresponding author.

References

E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness. Unsupervised label noise modeling and loss correction. In *ICML*, 2019.

D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. Kanwal, T. Maharaj, A. Fischer, A. Courville, and Y. Bengio. A closer look at memorization in deep networks. In *ICML*, 2017.

H. Bae, S. Shin, J. Jang, K. Song, and I. Moon. From noisy prediction to true label: Noisy prediction calibration via generative model. In *ICML*, 2022.

Y. Bai, E. Yang, B. Han, Y. Yang, J. Li, Y. Mao, G. Niu, and T. Liu. Understanding and improving early stopping for learning with noisy labels. In *NeurIPS*, 2021.

R. J. N. Baldock, H. Maennel, and B. Neyshabur. Deep learning through the lens of example difficulty. In *NeurIPS*, 2021.

T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *ISMIR*, 2011.

M. Boudiat, J. Rony, I. M Ziko, E. Granger, M. Pedersoli, P. Piantanida, and I. B. Ayed. A unifying mutual information view of metric learning: cross-entropy vs. pairwise losses. In *ECCV*, 2020.

T. Castells, P. Weinzaepfel, and J. Revaud. Superloss: A generic loss for robust curriculum learning. In *NeurIPS*, 2020.

C. de Vente, P. Vos, M. Hosseinzadeh, J. Pluim, and M. Veta. Deep learning regression for prostate cancer detection and grading in bi-parametric mri. *IEEE Transactions on Biomedical Engineering*, 68(2):374–383, 2021.

H. Doi, K. Z. Takahashi, H. Yasuoka, J. Fukuda, and T. Aoyagi. Regression analysis for predicting the elasticity of liquid crystal elastomers. *Scientific Reports*, 12(19788), 2022.

Y. Dukler, B. Bowman, A. Achille, A. Golatkar, A. Swaminathan, and S. Soatto. Safe: Machine unlearning with shard graphs. *arXiv preprint arXiv: 2304.13169*, 2023.

W. Fedus, B. Zoph, and N. Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018.

J. Gao, J. Wang, S. Dai, L. J. Li, and R. Nevatia. Note-rcnn: Noise tolerant ensemble rcnn for semi-supervised object detection. In *ICCV*, 2019.

W. Gao, B. Yang, and Z. Zhou. On the resistance of nearest neighbor to random noisy labels. *arXiv preprint arXiv:1607.07526*, 2016.

Z. Gao, S. Cheng, R. He, Z. Xie, H. Zhao, Z. Lu, and T. Xiang. Compressing deep neural networks by matrix product operators. In *Physical Review Research*, 2020.

Z. Gao, P. Liu, W. X. Zhao, Z. Lu, and J. Wen. Parameter-efficient mixture-of-experts architecture for pre-trained language models. In *Proceedings of the 29th International Conference on Computational Linguistics*, 2022.

B. Garg and N. Manwani. Robust deep ordinal regression under label noise. In *Asian Conference on Machine Learning*, 2020.

X. Geng and Y. Xia. Head pose estimation based on multivariate label distribution. In *CVPR*, 2014.

A. Gibbons, editor. *Algorithmic Graph Theory*. Cambridge University Press, London, England, 1985.

Y. Gong, G. Mori, and F. Tung. Ranksim: Ranking similarity regularization for deep imbalanced regression. In *ICML*, 2022.

Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko. Revisiting deep learning models for tabular data. In *NeurIPS*, 2021.

L. Grinsztajn, E. Oyallon, and G. Varoquaux. Why do tree-based models still outperform deep learning on tabular data? In *NeurIPS Track Datasets and Benchmarks*, 2022.

A. Grønlund, L. Kamma, K. G. Larsen, A. Mathiasen, and J. Nelson. Margin-based generalization lower bounds for boosted classifiers. In *NeurIPS*, 2019.

A. Grønlund, L. Kamma, and K. G. Larsen. Near-tight margin-based generalization bounds for support vector machines. In *ICML*, 2020.

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

B. Han, G. Niu, X. Yu, Q. Yao, M. Xu, I. W. Tsang, and M. Sugiyama. Sigua: Forgetting may make learning with noisy labels more robust. In *ICML*, 2020.

J. He, J. Qiu, A. Zeng, Z. Yang, J. Zhai, and J. Tang. Fastmoe: A fast mixture-of-expert training system. *arXiv preprint arXiv:2103.13262*, 2021.

K. A. Heller and Z. A. Ghahramani. Nonparametric bayesian approach to modeling overlapping clusters. In *Artificial Intelligence and Statistics*, 2007a.

Katherine A. Heller and Zoubin Ghahramani. A nonparametric bayesian approach to modeling overlapping clusters. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2, pages 187–194. PMLR, 2007b.

D. Hendrycks, M. Mazeika, D. Wilson, and K. Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. In *NeurIPS*, 2018.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. In *Neural Computation*, 2002.

R. Hirk, K. Hornik, and L. Vana. Multivariate ordinal regression models: an analysis of corporate credit ratings. *Statistical Methods & Applications*, 28:507–539, 2019.

W. Hu, Z. Li, and D. Y. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *ICLR*, 2020.

L. Huang, C. Zhang, and H. Zhang. Self-adaptive training: beyond empirical risk minimization. In *NeurIPS*, 2020.

Z. Huang, J. Zhang, and H. Shan. Twin contrastive learning with noisy labels. In *CVPR*, 2023.

S. C. H. Hoi J. Li, C. Xiong. Learning from noisy data with robust representation learning. In *ICCV*, 2021.

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Comput*, 3:79–87, 1991.

L. Jiang, Z. Zhou, T. Leung, L. Li, and L. Fei-Fei. Mentornet:learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, 2018.

N. Karim, M. N. Rizve, N. Rahnavard, A. Mian, and M. Shah. Unicon: Combating label noise through uniform selection and contrastive learning. In *CVPR*, 2022.

T. Kim, J. Ko, S. Cho, J. Choi, and S. Yun. Fine samples for learning with noisy labels. In *NeurIPS*, 2021.

Y. J. Kim, A. A. Awan, A. Muzio, A. Salinas, L. Lu, A. Hendy, S. Rajbhandari, Y. He, and H. H. Awadalla. Scalable and efficient moe training for multitask multilingual models. *arXiv preprint arXiv:2109.10465*, 2023.

M. Kimura, T. Nakamura, and Y. Saito. Shift15m: Multiobjective large-scale fashiondataset with distributional shifts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, 2021.

D. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015.

S. M. Kye, K. Choi, J. Yi, and B. Chang. Learning with noisy labels by efficient transition matrix estimation to combat label miscorrection. In *ECCV*, 2022.

K. Lee, X. He, L. Zhang, and L. Yang. Cleannet: Transfer learning for scalable image classfier training with label noise. In *CVPR*, 2018.

K. Lee, S. Yun, K. Lee, H. Lee, B. Li, and J. Shin. Robust inference via generative classifiers for handling noisy labels. In *ICML*, 2019.

D. Lepikhin, H. J. Lee, Y. Xu, D. Chen, O. Firat, and Y. Huang. Gshard: Scaling giant models with conditional computation and automatic sharding. In *ICLR*, 2021.

M. Lewis, S. Bhosale, T. Dettmers, N. Goyal, and L. Zettlemoyer. Base layers: Simplifying training of large, sparse models. *arXiv preprint arXiv:2103.16716*, 2021.

J. Li, R. Socher, and S. C. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICLR*, 2020a.

J. Li, C. Xiong, R. Socher, and S. Hoi. Towards noise-resistant object detection with noisy annotations. *arXiv preprint arXiv: 2003.01285*, 2020b.

J. Li, G. Li, F. Liu, and Y. Yu. Neighborhood collective estimation for noisy label identification and correction. In *ECCV*, 2022a.

S. Li, X. Xia, S. Ge, and T. Liu. Selective-supervised contrastive learning with noisy labels. In *CVPR*, 2022b.

S. Li, X. Xia, H. Zhang, Y. Zhan, S. Ge, and T. Liu. Estimating noise transition matrix with label correlations for noisy multi-label learning. In *NeurIPS*, 2022c.

W. Li, J. Lu, J. Feng, C. Xu, J. Zhou, and Q. Tian. Bridgenet: A continuity-aware probabilistic network for age estimation. In *CVPR*, 2019.

K. Lim, N. H. Shin, Y. Y. Lee, and C. S. Kim. Order learning and its application to age estimation. In *ICLR*, 2020.

C. Liu, K. Wang, H. Lu, Z. Cao, and Z. Zhang. Robust object detection with inaccurate bounding boxes. In *ECCV*, 2022.

S. Liu, J. Niles-Weed, N. Razavian, and C. Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. In *NeurIPS*, 2020.

Y. Liu, K. Duffy, J. G. Dy, and A. R. Gaunguly. Explainable deep learning for insights in el niño and river flows. *Nature Communications*, 14(339), 2023.

I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

J. Ma, Y. Ushiku, and M. Sagara. The effect of improving annotation quality on object detection datasets: A preliminary study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, 2022.

X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. M. Erfani, S. T. Xia, S. Wijewickrema, and J. Bailey. Dimensionality-driven learning with noisy labels. In *ICML*, 2018.

J. Mao, Q. Yu, Y. Yamakata, and K. Aizawa. Noisy annotation refinement for object detection. *British Machine Vision Conference*, 2021.

S. Masoudnia and R. Ebrahimpour. Mixture of experts: a literature survey. In *Artificial Intelligence Review*, 2014.

A. K. Menon, A. S. Rawat, S. J. Reddi, and S. Kumar. Can gradient clipping mitigate label noise? In *ICLR*, 2020.

B. Mirzasoleiman, K. Cao, and J. Leskovec. Coresets for robust training of neural networks against noisy labels. In *NeurIPS*, 2020.

K. H. Monfared and S. Mallik. Spectral characterization of matchings in graphs. *Linear Algebra and its Applciations*, 496(1):234–778, 2016.

Z. Niu, M. Zhou, X. Gao, and G. Hua. Ordinal regression with a multiple output cnn for age estimation. In *CVPR*, 2016.

D. Ortego, E. Arazo, P. Albert, N. E. O'Connor, and K. McGuinness. Multi-objective interpolation training for robustness to label noise. In *CVPR*, 2021.

P. Ostyakov, E. Logacheva, R. Suvorov, V. Aliev, G. Sterkin, O. Khomenko, and S. I. Nikolenko. Label denoising with large ensembles of heterogeneous neural networks. In *ECCV*, 2018.

S. Papadopoulos, C. Koutlis, S. Papadopoulos, and I. Kompatsiaris. Multimodal quasi-autoregression: forecasting the visual popularity of new fashion products. *International Journal of Multimedia Information Retrieval*, 11:717–729, 2022.

G. Patrini, A. Rozza, A. Menon, R. Nock, and L. Qu. Making deep neural networks robust to label noise: a loss correction approach. In *CVPR*, 2017.

G. Pleiss, T. Zhang, E. R. Elenberg, and K. Q. Weinberger. Identifying mislabeled data using the area under the margin ranking. In *NeurIPS*, 2020.

S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *ICLR workshop*, 2015.

M. Ren, W. Zeng, B. Yang, and R. Urtasun. Learning to reweight examples for robust deep learning. In *ICML*, 2018.

R. Rothe, R. Timofte, and L. V. Gool. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision*, 126(2-4):144–157, 2018.

Rasmus Rothe, Radu Timofte, and Luc Van Gool. Dex: Deep expectation of apparent age from a single image. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 10–15, 2015.

Amir M Sarfi, Zahra Karimpour, Muawiz Chaudhary, Nasir M Khalid, Mirco Ravanelli, Sudhir Mudur, and Eugene Belilovsky. Simulated annealing in early layers leads to better generalization. In *CVPR*, pages 20205–20214, 2023.

M. Schubert, T. Riedlinger, K. Kahl, D. Kröll, S. Schoenen, S. Šegvić, and M. Rottmann. Identifying label errors in object detection datasets by loss inspection. *arXiv preprint arXiv: 2303.06999*, 2023.

D. Shah, Z. Y. Xue, and T. M. Aamodt. Label encoding for regression networks. *arXiv preprint arXiv:2212.01927*, 2022.

A. Sharkey and N. Sharkey. Combining diverse neural nets. In *The Knowledge Engineering Review*, 1997.

J. Shawe-Taylor and N. Cristianini. Robust bounds on generalization from the margin distribution. 1998.

Y. Shen and S. Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In *ICML*, 2019.

Y. Shen, R. Ji, Z. Chen, X. Hong, F. Zheng, J. Liu, M. Xu, and Q. Tian. Noise-aware fully webly supervised object detection. In *CVPR*, 2020.

N. Shin, S. Lee, and C. Kim. Moving window regression: a novel approach to ordinal regression. In *CVPR*, 2022.

H. A. Sia, R. Baldrich, M. Vanrell, and D. Samaras. Light direction and color estimation from single image with deep regression. *arXiv preprint arXiv:2009.08941*, 2020.

H. Song, M. Kim, and J. Lee. Selfie: Refurbishing unclean samples for robust deep learning. In *ICML*, 2019.

Cory Stephenson, Abhinav Ganesh, Yue Hui, Hanlin Tang, SueYeon Chung, et al. On the geometry of generalization and memorization in deep neural networks. In *ICLR*, 2021.

H. Su, J. Deng, and L. Fei-Fei. Crowdsourcing annotations for visual object detection. In *HCOMP@AAAI*, 2012.

S. Tanaka, N. Kadoya, Y. Sugai, M. Umeda, M. Ishizawa, Y. Katsuta, K. Ito, K. Takeda, and K. Jingu. A deep learning-based radiomics approach to predict head and neck tumor regression for adaptive radiotherapy. *Scientific Reports*, 12(8899), 2022.

Wenhai Wan, Xinrui Wang, Ming-Kun Xie, Shao-Yuan Li, Sheng-Jun Huang, and Songcan Chen. Unlocking the power of open set: A new perspective for open-set noisy label learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 15438–15446, 2024.

Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey. Symmetric cross entropy for robust learning with noisy labels. In *ICCV*, 2019.

Y. Wang, X. Sun, and Y. Fu. Scalable penalized regression for noise detection in learning with noisy labels. In *CVPR*, 2022.

Z. Wang, G. Hu, and Q. Hu. Training noise-robust deep neural networks via meta-learning. In *CVPR*, 2020.

H. Wei, L. Feng, X. Chen, and B. An. Combating noisy labels by agreement: A joint training method with co-regularization. In *CVPR*, 2020.

Hongxin Wei, Lue Tao, Renchunzi Xie, and Bo An. Open-set label noise can improve robustness against inherent label noise. *NeurIPS*, 34:7978–7992, 2021.

Cheng Wen-Huang, Song Sijie, Chen Chieh-Yun, Hidayati Shintami Chusnul, and Liu Jiaying. Fashion meets computer vision: A survey. *ACM Computing Surveys*, 2021.

P. Wu, S. Zheng, M. Goswami, D. N. Metaxas, and C. Chen. A topological filter for learning with label noise. In *NeurIPS*, 2020a.

Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *KDD*, 2020b.

X. Xia, T. Liu, B. Han, N. Wang, M. Gong, H. Liu, G. Niu, D. Tao, and M. Sugiyama. Part-dependent label noise: Towards instance-dependent label noise. In *NeurIPS*, 2020.

X. Xia, T. Liu, B. Han, C. Gong, N. Wang, Z. Ge, and Y. Chang. Robust early-learning: Hindering the memorization of noisy labels. In *ICLR*, 2021.

X. Xia, T. Liu, B. Han, M. Gong, J. Yu, G. Niu, and M. Sugiyama. Sample selection with uncertainty of losses for learning with noisy labels. In *ICLR*, 2022.

S. Yang, E. Yang, B. Han, Y. Liu, M. Xu, G. Niu, and T. Liu. Estimating instance-dependent label-noise transition matrix using dnns. In *ICML*, 2022a.

Y. Yang, K. Zha, Y. Chen, and H. Wang D. Katabi. Delving into deep imbalanced regression. In *ICML*, 2022b.

H. Yao, Y. Wang, L. Zhang, J. Zou, and C. Finn. C-mixup: Improving generalization in regression. In *NeurIPS*, 2022.

Y. Yao, T. Liu, B. Han, M. Gong, J. Deng, G. Niu, and M. Sugiyama. Dual t: Reducing estimation error for transition matrix in label-noise learning. In *NeurIPS*, 2020.

K. Yi and J. Wu. Probabilistic end-to-end noise correction for learning with noisy labels. In *CVPR*, 2019.

L. Yi, S. Liu, Q. She, A. McLeod, and B. Wang. On learning contrastive representations for learning with noisy labels. In *CVPR*, 2022.

L. Yiming, S. Jie, W. Yujiang, and P. Maja. Fp-age: Leveraging face parsing attention for facial age estimation in the wild. *arXiv*, 2021.

S. E. Yuksel, J. N. Wilson, and P. D. Gader. Twenty years of mixture of experts. In *Transactions on neural networks and learning systems*, 2012.

T. Zadouri, A. Üstün, A. Ahmadian, B. Ermis, A. Locatelli, and S. Hooker. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. *arXiv preprint arXiv:2309.05444*, 2023.

S. Zang, M. Ding, D. B. Smith, P. Tyler, T. Rakotoarivelo, and M. Ali Kâafar. The impact of adverse weather conditions on autonomous vehicles: How rain, snow, fog, and hail affect the performance of a self-driving car. *IEEE Vehicular Technology Magazine*, 14:103–111, 2019.

K. Zha, P. Cao, Y. Yang, and D. Katabi. Supervised contrastive regression. *arXiv preprint arXiv:2210.01189*, 2022.

C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017a.

H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

L. Zhang, C. Aggarwal, and G. Qi. Stock price prediction via discovering multi-frequency trading patterns. In *KDD*, 2017b.

S. Zhang, L. Yang, M. B. Mi, X. Zheng, and A. Yao. Improving deep regression with ordinal entropy. In *ICLR*, 2023.

X. Zhang, Z. Liu, K. Xiao, T. Shen, J. Huang, W. Yang, D. Samaras, and X. Han. Codim: Learning with noisy labels via contrastive semi-supervised learning. *arXiv preprint arXiv: 2111.11652*, 2021a.

Y. Zhang, H. Sun, G. Gao, L. Shou, and D. Wu. Developing spatio-temporal approach to predict economic dynamics based on online news. *Scientific Reports*, 12(16158), 2022.

Z. Zhang and M. Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *NeurIPS*, 2018.

Z. Zhang, Y. Lin, Z. Liu, P. Li, M. Sun, and J. Zhou. Moefication: Conditional computation of transformer models for efficient inference. *arXiv preprint arXiv:2110.01786*, 2021b.

Z. Zhifei, S. Yang, and Q. Hairong. Age progression regression by conditional adversarial autoencoder. In *CVPR*, 2017.

Hattie Zhou, Ankit Vani, Hugo Larochelle, and Aaron Courville. Fortuitous forgetting in connectionist networks. In *ICLR*, 2022.

M. Zhou, Y. Xu, L. Ma, and S. Tian. On the statistical errors of radar location sensor networks with built-in wi-fi gaussian linear fingerprints. *Sensors (Basel, Switzerland)*, 12:3605 – 3626, 2012.

Z. Zhu, J. Wang, and Y. Liu. Beyond images: Label noise transition matrix estimation for tasks with lower-quality features. In *ICML*, 2022.

Chen-Chen Zong, Ye-Wen Wang, Ming-Kun Xie, and Sheng-Jun Huang. Dirichlet-based prediction calibration for learning with noisy labels. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(15):17254–17262, 2024.

B. Zoph, I. Bello, S. Kumar, N. Du, Y. Huang, J. Dean, N. Shazeer, and W. Fedus. Designing effective sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022.

S. Zuo, X. Liu, J. Jiao, Y. J. Kim, H. Hassan, R. Zhang, T. Zaho, and J. Gao. Taming sparsely activated transformers with stochastic experts. *CoRR CoRR:2110.04260*, 2021.

A Appendix: Table of Contents

The Appendix enlists the following additional materials.

B Limitation

A key limitation of ConFrag lies in its foundational reliance on the Mixture of Experts (MoE) model [Jacobs et al., 1991]. Specifically, integrating MoEs with deep learning introduces notable scalability challenges, both computationally and in memory usage [Zuo et al., 2021, Zoph et al., 2022, Zhang et al., 2021b]. To address the memory concern, ConFrag currently employs more compact feature extractors. Nevertheless, a prominent inefficiency stems from expert redundancy in MoEs' parameters [Zuo et al., 2021]. Some approaches to mitigate this include distilling into sparse MoE models, employing pruning, and subsequently compressing to decrease parameter size [Kim et al., 2023, Fedus et al., 2021]. There are also emerging strategies centered on parameter sharing, leveraging matrix product operators (MPO) decomposition [Gao et al., 2020, 2022] and parameter-efficient fine-tuning [Zadouri et al., 2023]. Of these, we believe the avenue of parameter sharing holds special promise when combined with ConFrag; the inherent positive feature correlation in regression problems amplifies the advantages of this approach. Also, as in MoEs, ConFrag introduces new hyperparameter, the number of experts (the number of fragments F in ConFrag's case).

In its current form, ConFrag facilitates simultaneous training of both the feature extractors and the subsequent regression task, either on a per-batch or per-epoch basis. However, a wealth of research exists that could further optimize ConFrag's scalability. These span from improving training efficiency [He et al., 2021, Zoph et al., 2022, Lepikhin et al., 2021, Lewis et al., 2021] to enhancing inference capabilities [Zhang et al., 2021b, Fedus et al., 2021].

C Broader Impacts

In the era of deep learning, the need for large datasets increases, yet it is expensive to obtain large dataset with high-quality annotated labels. An alternative solution is to collect labels using automated labeling methods, such as web crawling. However, these methods inevitably introduce noisy labels.

This work proposes a method for mitigating the negative effect of such label noise in regression, which can save time and money spent on collecting high-quality labels for many applications, bringing positive impact on science, society, and economy. However, since the method reduces the need for accurate labeling, it may have potential negative effect on the salaries of label workers.

D Theory of ConFrag

We present several theoretical justifications that enhance the performance of ConFrag.

D.1 Classification versus Regression for Feature Learning

During the learning process, deep neural networks aim to maximize the mutual information between the learned representation, denoted as Z, and the target variable, denoted as Y. The mutual information between these two variables can be defined as $I(Z;Y) = H(Z) - H(Z|Y)$. A high value of $I(Z;Y)$ is indicative of a high marginal entropy $H(Z)$. Achieving this dual objective is accomplished in classification [Boudiat et al., 2020].

However, Zhang et al. [2023] have shown that regression primarily focuses on minimizing $H(Z|Y)$ while disregarding $H(Z)$. This results in a relatively lower marginal entropy for the learned representation Z and ultimately leads to performance deficits in comparison to classification.

To experimentally show that this theoretical result also applies to ConFrag, we replace classification-based expert feature extractor learning with regression-based one, where each expert feature extractor is trained with regression loss on its respective fragment pair dataset. We name this variant ConFrag-R. In ConFrag-R, self-agreement is defined using distances to the mean of each fragment in the contrasting pair (f, f^+):

$$\alpha_f^{\text{self}} = \left[|\bar{Y}_f - h(x;\theta_{f,f^+})| < |\bar{Y}_{f^+} - h(x;\theta_{f,f^+})| \right], \tag{6}$$

where $\bar{Y}_f$ is the average of fragment f's labels, and $h_R(\cdot)$ is the regression function output. As in ConFrag, ConFrag-R also utilizes K-nearest neighbor-based classification for computing another variant of self-agreement. The results in Table 4 show that using classification for feature learning outperforms using regression (ConFrag-R) in all datasets.

D.2 Fragmentation and Neighborhood Jittering

ConFrag operates by partitioning data samples into fragments and leveraging trained feature extractors for sample selection through collective modeling. We conceptualize this as a Mixture-of-Experts (MoE) model, wherein individual experts specialize in specific problem subspaces through data partitioning [Yuksel et al., 2012, Masoudnia and Ebrahimpour, 2014]. MoEs possess theoretically advantageous properties with respect to computational scalability and reduction of output variance [Yuksel et al., 2012], contributing to the enhancements observed in ConFrag. It is noteworthy that since each network is trained on a distinct training set, MoE effectively mitigates concurrent failures, thereby preventing error propagation among networks and ultimately improving the generalization performance of ConFrag as well [Sharkey and Sharkey, 1997].

Additionally, our Neighborhood Jittering leads to a Partially Overlapping Mixture Model [Heller and Ghahramani, 2007a], theoretically enabling the modeling of significantly richer and more

Table 4: **Comparison between ConFrag and ConFrag-R: Mean Relative Absolute Error (%)** to the noise-free trained model on the AFAD-B, IMDB-Clean-B, SHIFT15M-B, and MSD-B dataset. Lower is better. The results are the mean of three random seed experiments.

	AFAD-B						IMDB-Clean-B					
	symmetric				Gaussian		symmetric				Gaussian	
noise rate	20	40	60	80	30	50	20	40	60	80	30	50
Vanilla	9.37	20.27	30.65	43.09	28.77	39.03	16.18	32.05	53.13	76.35	26.89	50.28
ConFrag-R	4.97	13.93	27.85	37.19	21.93	33.90	8.74	22.73	44.29	68.14	21.74	46.93
ConFrag	**2.74**	**8.16**	**15.91**	**34.42**	**17.49**	**27.31**	**5.08**	**12.64**	**27.26**	**61.24**	**15.70**	**33.36**

	SHIFT15M-B						MSD-B					
	symmetric				Gaussian		symmetric				Gaussian	
noise rate	20	40	60	80	30	50	20	40	60	80	30	50
Vanilla	9.11	17.96	27.02	36.34	6.54	15.16	8.23	18.43	31.67	45.85	6.96	15.74
ConFrag-R	4.18	9.59	16.21	25.76	4.96	10.90	0.77	5.68	13.63	30.05	2.79	6.87
ConFrag	**2.46**	**6.18**	**10.68**	**19.04**	**3.66**	**8.09**	**0.57**	**4.94**	**11.22**	**23.41**	**2.39**	**6.49**

intricate hidden representations by accommodating multi-cluster membership, ultimately enhancing the selection and overall performance of ConFrag.

D.3 Prediction Depth Analysis

Prediction depth [Baldock et al., 2021] of an example refers to the earliest layer where the layer-wise K-nearest neighbor probes of the layer and all the subsequent layers are the same as the model prediction. In other words, low prediction depth means that the example is easily distinguishable in early layers. For example, a prediction depth of zero means that data can be predicted at the input level only based on its distances to other data. Low prediction depth is positively correlated with better prediction consistency, lower learning difficulty, and larger margin. Due to these traits, some previous works aim at reducing the prediction depth during training for better generalization performance [Zhou et al., 2022, Sarfi et al., 2023].

While prediction depth is initially designed as a measure of example difficulty, the mean prediction depth of the dataset can also be used as a measure of dataset difficulty [Baldock et al., 2021]. Also, since early layers generalize while later layers memorize in deep learning [Stephenson et al., 2021], the low mean prediction depth of a dataset means it is more generalizable since fewer examples require memorization.

Table 5: **Comparison of mean prediction depths** of feature extractor learning tasks for all-frag, contrastive fragment pairing, and alternative fragmentation pairings when $F = 4$.

		IMDB-Clean-B	
Fragment pairing		No noise	Symmetric 40%
All-frag ([1-4])		6.6291	7.2452
Contrastive pairing	[1, 3]	3.7752	4.8116
	[2, 4]	3.7028	4.7869
Alternative pairing 1	[1, 4]	3.0822	4.3307
	[2, 3]	4.8215	5.3978
Alternative pairing 2	[1, 2]	4.8738	5.4741
	[3, 4]	4.7304	5.1828

Table 6: **Comparison of mean prediction depths** of feature extractor learning tasks for all-frag, contrastive fragment pairing, and alternative fragmentation pairings when $F = 6$.

		IMDB-Clean-B	
Fragment pairing		No noise	Symmetric 40%
All-frag ([1-6])		7.2689	7.8102
Contrastive pairing	[1, 4]	3.8635	4.8706
	[2, 5]	3.7668	4.6382
	[3, 6]	3.6840	4.5026
Alternative pairing 1	[1, 2]	4.8979	5.4790
	[3, 4]	5.0770	5.2453
	[5, 6]	4.8010	5.0605
Alternative pairing 2	[1, 6]	2.9685	4.1188
	[2, 5]	3.7668	4.6382
	[3, 4]	5.0770	5.2453

Tab. 5–6 show the mean prediction depth of all samples for each feature extractor learning task, with and without noise. The prediction depth for each task is measured using ResNet-18 trained for 20 epochs, at which the model achieves more than 99% training accuracy. Following Baldock et al. [2021], we use $K = 30$ for K-nearest neighbor probe and did not use data augmentation during training. The tables show that the binary classification tasks from contrastive pairing achieve much lower prediction depths than the multi-class classification tasks using all fragments (All-frag). Also, the mean and maximum prediction depths of contrastive pairing are lower than those of alternative pairings, explaining why contrastive pairing outperforms alternative pairings as shown in § 4.3. Note that the mean prediction depths of the binary classification tasks correlate with the distance between fragments: the larger the distance, the lower the prediction depth tends to be.

Fig. 7– 8 show the distribution of prediction depth for each task when $F = 4$ and $F = 6$. Without noise, about 40% of data in each contrastive fragment pair has a prediction depth of zero, indicating that two fragments are already much separated at input level. Even with 40% symmetric noise, more than 25% of data in each contrastive fragment pair has a prediction depth of zero. Meanwhile, the most frequent prediction depth when using all fragments is nine, which means that most data can only be predicted at the last hidden feature level.

While the prediction depths of alternative pairings are lower than those of using all fragments, we observe that they tend to perform worse as shown in Appendix G.4. We suspect that this is due to the limitation of mixture models that the individual expert feature extractor may not fully benefit from the full dataset as they model their own disjoint subset.

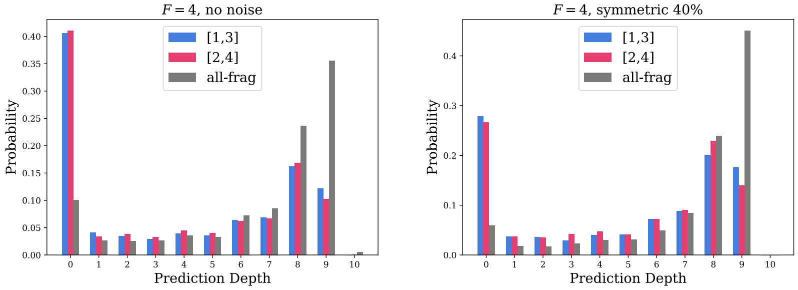


Figure 7: **Probability of prediction depth** for examples in contrastive pairing and all-frag ($F = 4$).

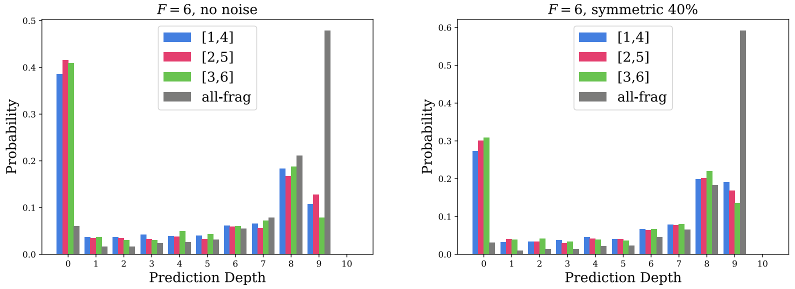


Figure 8: **Probability of prediction depth** for examples in contrastive pairing and all-frag ($F = 6$).

E Extended Related Work

E.1 Continuously Ordered Correlation of Labels and Features

One distinctive characteristic of regression problems is their continuous label space, implying a high likelihood of correlation between regions within the feature and label spaces [Yang et al., 2022b, Gong et al., 2022, Zha et al., 2022].

Recent research has extensively explored these characteristics, encompassing issues such as label imbalance [Yang et al., 2022b, Gong et al., 2022], age estimation [Li et al., 2019], contrastive learning [Zha et al., 2022], and mixup regularization [Yao et al., 2022].

Yang et al. [2022b] propose label and feature distribution smoothing based on their similarity, while Gong et al. [2022] introduce a regularization term aimed at aligning the rankings of feature-space and label-space neighbors. Zha et al. [2022] employ supervised contrastive learning with a pairing technique based on label distances in mini-batches. To adapt MixUp [Zhang et al., 2018] for regression tasks, Yao et al. [2022] recommend interpolating proximal samples within the label space with a higher probability.

Ordinal regression, also known as ranking learning, pertains to predicting ordinal labels based on input data. It is noteworthy that ordinal regression methods are adaptable for regression tasks due to the inherent numerical ordering within scalar label spaces. Past studies in ordinal regression have successfully addressed various regression challenges, including facial age estimation [Niu et al., 2016, Shin et al., 2022], monocular depth estimation [Fu et al., 2018], and credit rating [Hirk et al., 2019]. Some of these methods share common characteristics with our approach, as they discretize continuous labels, effectively converting regression tasks into classification problems [Niu et al., 2016, Fu et al., 2018, Shah et al., 2022]. Within the framework of ordinal regression, Garg and Manwani [2020] propose a loss correction method by estimating the noise transition matrix.

It is important to note that among the previously mentioned methods, only Yao et al. [2022] and Garg and Manwani [2020] can effectively address noisy label regression problems without the need for additional techniques. Additionally, Wang et al. [2022] enhance the scalability of their approach by grouping dissimilar classes within the feature space. Our work considers the continuity of labels and features and their correlation in fragmenting and grouping data. This approach allows each component to learn distinguishable features and improve sample selection capabilities.

E.2 Noisy Label in Object Detection

Due to the abundance of research on object detection tasks, with bounding box localization being a prominent example of regression tasks, we have explored the issue of noisy regression within the context of object detection. In particular, obtaining accurate annotations for object detection is a resource-intensive task, often constrained by limited time, a small number of annotators, or reliance on machine-generated annotations. These constraints frequently result in label noise, represented as incorrect class assignments or inaccurate bounding box locations.

Various strategies have been developed to address the issue of noisy labels in object detection. To correct inaccurate bounding box locations, Li et al. [2020b] leverage the discrepancy between two classification heads by emphasizing the objectness of the region. Liu et al. [2022] generates object bags using the classifier as guidance, Mao et al. [2021] employs center-matching correction, and Schubert et al. [2023] drop instances with high region proposal loss on an instance-wise basis. In scenarios where image-level annotations are available, Gao et al. [2019] employs ensemble learning with two classification heads and a distillation head, while Shen et al. [2020] decomposes the problem into foreground and background noise, employing residual learning and bagging-mixup learning.

We also explored the possibility of applying object detection techniques to noisy labeled regression. However, our analysis revealed that these methods are not well-suited for the broader regression task. Specifically, Liu et al. [2022], Schubert et al. [2023], Mao et al. [2021] utilize region proposal networks to generate bounding box proposals. They leverage these proposals to selectively choose clean labels or re-weight the training samples. However, because this approach necessitates an auxiliary model in the proposal generation process, it cannot be directly applied in the context of regression tasks.

Additionally, Li et al. [2020b], Liu et al. [2022], Schubert et al. [2023], Gao et al. [2019] employ the object detector's classifier to update or assess the quality of bounding boxes. By evaluating the confidence or consistency of the bounding box through the classification output, this approach helps mitigate the impact of noisy labels. However, implementing a similar approach in the context of regression tasks would require the inclusion of an auxiliary co-trained task.

E.3 Transition Matrix based Methods

Methods based on transition matrices constitute one of the primary approaches for addressing the issue of noisy labels.

Driven by the observation that the clean class posterior, denoted as $p(y^{\text{gt}}|x)$, can be inferred from the transition probability and the noisy class posterior, $p(y|x) = T(y|y^{\text{gt}})p(y^{\text{gt}}|x)$, the modification of the loss function enables the construction of a risk-consistent estimator using the estimated transition matrix [Yao et al., 2020].

There are many approaches aiming to enhance the estimation of the transition matrix. These include factorizing it into the product of two matrices by introducing an intermediate class [Yao et al., 2020], training the Bayes label transition network [Yang et al., 2022a], learning the transition matrix within a meta-learning framework [Wang et al., 2020], down-weighting less informative features based on f-mutual information [Zhu et al., 2022], and adopting a two-head architecture. The latter involves a noisy classifier for simultaneous transition matrix estimation and a clean classifier for statistically consistent training [Kye et al., 2022].

Moreover, Xia et al. [2020] explores the utilization of part-dependent transition matrices, combining them to approximate the instance-dependent transition matrix.

In an extended context, Li et al. [2022c] broadens the problem to include noisy multi-label learning and suggests considering label correlations.

E.4 Combination with Contrastive Learning

Incorporating unsupervised learning methods proves effective in alleviating label noise, prompting the integration of noisy label mitigation techniques with unsupervised learning, particularly contrastive learning.

Zhang et al. [2021a] show that the combination of contrastive loss and semi-supervised loss yields successful mitigation of the noisy label problem.

Beyond the application of contrastive learning, other approaches involve selecting confidence pairs and confidence samples [Li et al., 2022b], leveraging clean probability estimation derived from the relationship between representation clusters and labels [Huang et al., 2023], employing class prototypes for weakly-supervised loss [J. Li, 2021], and implementing soft-labeling based on the relation between representations and labels [Ortego et al., 2021].

Additionally, an approach introduces a contrastive regularization function aimed at preventing adverse effects stemming from noisy labels [Yi et al., 2022].

Table 7: **Dataset Statistics** on the six newly curated balanced datasets for regression: AFAD-B [Niu et al., 2016], IMDB-Clean-B [Yiming et al., 2021], IMDB-WIKI-B [Rothe et al., 2018], UTKFace-B [Zhifei et al., 2017], SHIFT15M-B [Kimura et al., 2021], MSD-B [Bertin-Mahieux et al., 2011].

Dataset	range	train	valid	test	total
AFAD-B	[15, 40]	27647	1627	3252	32526
IMDB-Clean-B	[15, 66]	44200	2600	5200	52000
IMDB-WIKI-B	[15, 65]	42500	2500	5000	50000
UTKFace-B	[1, 70]	10467	386	787	11640
SHIFT15M-B	[0, 40000]	273417	16080	32180	321677
MSD-B	[1956, 2010]	25218	1512	2970	29700

F Experiment Details

F.1 Dataset Curation Details

Table 7 provides a comprehensive overview of the statistics for the six benchmark datasets meticulously curated for the task of noisy label regression. Detailed descriptions of the dataset tailoring process are presented below for clarity.

Age prediction datasets (IMDB-Clean-B, AFAD-B, IMDB-WIKI-B, UTKFace-B): These datasets are harmonized by achieving a balance across distinct age values. This equilibrium is established using a bin sample count threshold (clip value) of 1000 for IMDB-Clean-B and IMDB-WIKI-B, 1251 for AFAD-B, and 200 for UTKFace-B. Image inputs are resized to dimensions of (128×128). For the regression task, we consistently employ a ResNet-50 backbone across all models.

SHIFT15M-B: Achieving data balance in this dataset involves a two-step process. First, the label space is binned based on a price threshold of ¥2000. Subsequently, data points exceeding the maximum price of ¥40000 are clipped to remove outliers. The binning threshold is set at 16084 sample counts to further ensure balanced representation. To standardize the label currency, it is pegged to the U.S. dollar, referencing exchange rates from 2010 to 2020, which coincides with the period when the original clothing item data is collected. Notably, this dataset is provided as the penultimate feature of the ImageNet pre-trained VGG-16 model. Consequently, we opt for a three-layer MLP architecture with a hidden layer size of [2048, 1024, 512], aligning with recommendations from Papadopoulos et al. [2022] and Kimura et al. [2021].

MSD-B: Achieving balance in the Million Song Dataset involves setting a threshold of 550 samples per year. For all regression models in this context, we adopt a regression backbone rooted in the tabular ResNet structure proposed by Gorishniy et al. [2021], featuring a hidden dimension of 467.

Licenses of existing datasets. IMDB-Clean dataset [Yiming et al., 2021] is under MIT license.[3] SHIFT15M dataset [Kimura et al., 2021] is under CC BY-NC 4.0 and MIT license.[4] MSD [Bertin-Mahieux et al., 2011] song year prediction dataset is under CC BY 4.0 license.[5] UTKFace dataset [Zhifei et al., 2017] is available for non-commercial research purposes only.[6] IMDB-WIKI dataset [Rothe et al., 2015, 2018] is available for academic research purpose only.[7] Unfortunately, the license of AFAD [Niu et al., 2016] dataset could not be found.[8]

F.2 Baselines Details

While numerous branches of noisy labeled learning have been explored for classification tasks, our focus in this study centers on the challenging domain of noisy label regression. To comprehensively investigate this task, we have conducted an extensive review of the various branches and have selected

[3] `https://github.com/yiminglin-ai/imdb-clean`

[4] `https://github.com/st-tech/zozo-shift15m`

[5] `https://archive.ics.uci.edu/dataset/203/yearpredictionmsd`

[6] `https://susanqq.github.io/UTKFace/`

[7] `https://data.vision.ee.ethz.ch/cvl/rrothe/imdb-wiki/`

[8] `https://github.com/John-niu-07/tarball`

a set of fourteen baselines that are adaptable to regression. It is worth noting that C-Mixup [Yao et al., 2022] was originally proposed as a regression baseline. In the following section, we provide an overview of these selected baselines, offering a broad coverage of diverse approaches to address the noisy label regression problem. Additionally, we present detailed descriptions of the experimental settings for each baseline.

1. D2L [Ma et al., 2018] for intrinsic dimension exploration. Following the paper, we set $k = 20$ and $m = 10$ for Local Intrinsic Dimensionality (LID) estimation and set the LID estimation window as five following the official implementation.
2. CDR [Xia et al., 2021] for model weight parameter selection, and RDI [Hu et al., 2020] for regularizing the paramter distance from the initialization. At RDI, we use search space $\lambda \in [0.25, 0.5, 1, 2, 4, 8]$.
3. C-Mixup [Yao et al., 2022] for regularization via continuous mixup. C-Mixup-batch is used in all experiments because of the excessive memory requirement for pairwise distance matrix P. We set the beta distribution variable α as 1.5. The bandwidth variable σ is searched over [0.01, 0.1, 1], following Yao et al. [2022].
4. SELFIE [Song et al., 2019] and AUX [Hu et al., 2020] for refurbishing. To apply SELFIE to the continuous label, we redefine the concept of uncertainty $F(x; q)$ and refurbished labels y^{refurb} with the mean and standard deviation.

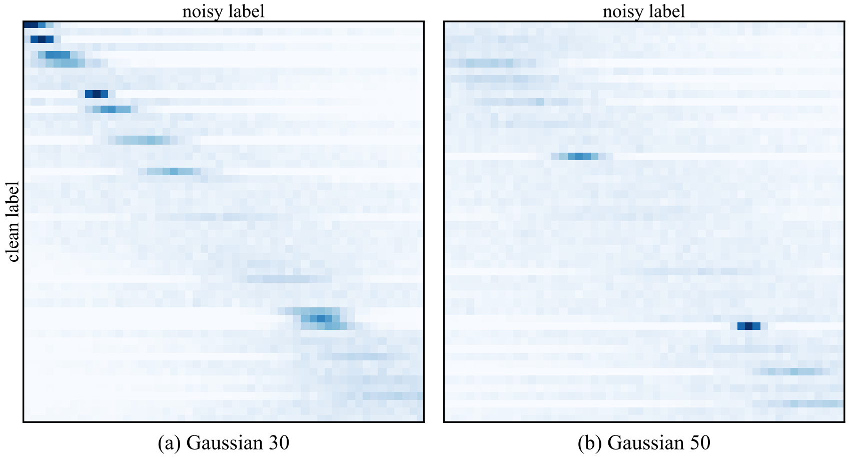
   $$F(x; q) = \frac{\sigma(H_x(q))}{(\max(Y) - \min(Y))} < \epsilon \tag{7}$$

   $$y^{refurb} = \mu(H_x(q)) \tag{8}$$

   where $H_x(q)$ is the prediction history of x from before q epochs, ϵ is the uncertainty threshold.

   For SELFIE, we train 1/4 of the total training epochs for the warm-up phase, following Song et al. [2019]. The variable q is searched over half of the warm-up epochs and around. The variable ϵ is searched over [0.05, 0.10, 0.15, 0.20], following Song et al. [2019].

   For AUX [Hu et al., 2020], we regularize the auxiliary variable by weight decay 0.0005, reducing the weight by 0.1 at 1/2 and 3/4 of the total training epochs. The learning rate of the auxiliary variable is set to 0.1 and 0.01. The variable λ is searched over [0.25, 0.5, 1, 2, 4, 8].
5. SPR [Wang et al., 2022] performs penalized regression for selection. It requires some adaptation to regression by ignoring the ℓ_q penalty as there is no longer a linearity gap between the scalar output and the final fully connected layer that requires reducing. Also, we use our fragmentation splits $\{4, 8\}$ to bin the regression data for SPR's parallel optimization.
6. Sigua [Han et al., 2020] and CNLCU-S/H [Xia et al., 2022] for small loss selection. For Sigua, we use $\delta(t) \in [0.3, 0.4]$ and $\gamma = 0.01$ and set T_k as 5% of the total training epochs. For CNLCU-S/H, we search σ and $\tau_{\min}$ in [0.01, 0.1, 1, 10] and set T_k as 5%.
7. BMM [Arazo et al., 2019] for selection based on beta mixture model fitting on the loss distribution. BMM does hard sampling and trains using the selected samples. DY-S is a dynamic soft loss. We implemented two versions; the first uses a convex combination as in Reed et al. [2015] $((1 - w)\tilde{y}^c - w\hat{y})^2$. Second, instead of bootstrapping, we dynamically weight the loss using the BMM probability to create a cost-sensitive loss, $(1 - w)\ell$. The w is the mixture clean probability, $\hat{y}$ is the model prediction, $\tilde{y}^c$ is the assigned noisy label, and ℓ is the loss.
8. SuperLoss [Castells et al., 2020] regularization parameter λ is searched over [0.01, 0.1, 1, 10] while τ uses an exponential running average with a fixed smoothing parameter $\alpha = 0.9$.
9. [Incompatible] CRUST [Mirzasoleiman et al., 2020] for clean coreset selection. It aims to select a coreset based on *class-wisely gradient clustering*. For regression, we initially viewed *all data as a single class* and proceeded with coreset selection, but the results were unsatisfactory. Therefore, we report results based only on the discretized version, demonstrating comparable performances. We select 1/2 of the total dataset as a coreset. The distance threshold in calculating clusters is searched over [1, 2, 4].



(a) Gaussian 30

(b) Gaussian 50

Figure 9: **Random Gaussian Noise.** (a) Gaussian noise injected from the uniformly sampled random standard deviation between $[1, 30]$. (b) Gaussian noise injected from uniformly sampled random standard deviation between $[1, 50]$.

10. [Incompatible] OrdRegr [Garg and Manwani, 2020] for loss correction. Since no official implementation is provided, we implemented it with cross-entropy loss for ordinal regression. Importantly, we failed to find accurate noise rate estimation using their suggested methods. Even when considering the transition matrix with the actual noise rate, the loss correction algorithm proved ineffective in our benchmark tests.

F.3 ConFrag Training Details

ConFrag employs the Cosine Annealing Learning rate [Loshchilov and Hutter, 2017] with a minimum learning rate of $\eta_{min} = 0$. The optimization is carried out using the Adam optimizer [Kingma and Ba, 2015]. For the K-nearest neighbors-based prediction, we experiment with various values of K, specifically choosing from the set $[3, 5, 7]$. The number of fragments, denoted as F, remains constant at four throughout all the experiments. To determine the buffer range for jittering, we conduct a search over values within the range $[0, 0.05, 0.1]$.

Some dataset-specific hyperparameters exist:

- Age prediction task datasets, IMDB-Clean-B [Yiming et al., 2021], AFAD-B [Niu et al., 2016], IMDB-WIKI-B [Rothe et al., 2018], and UTKFace-B [Zhifei et al., 2017], train for 120 epochs with a learning rate of 0.001. Each feature extractor employs the ResNet-18 architecture, which contains only 48% of the parameters found in ResNet-50, the architecture utilized for the regressor.
- Clothing price estimation task dataset SHIFT15M-B [Kimura et al., 2021] trains for 40 epochs with a learning rate of 0.0001. MLP with hidden dimensions [1024, 512, 256] is deployed for feature extractors, and the parameter size is 44% of the regressor.
- Music year production task dataset MSD-B [Bertin-Mahieux et al., 2011] trains for 20 epochs with a learning rate of 0.0001. Similar to the regression backbone, the feature extractor model is the tabular ResNet structure[Gorishniy et al., 2021], and the hidden dimension is reduced to 256.

F.4 Random Gaussian Noise

Fig. 9 illustrates the application of random Gaussian noise within the label space of IMDB-Clean-B [Yiming et al., 2021]. The procedure for injecting noise is akin to the approach employed by Yao et al. [2022], where Gaussian noise is applied to every unique label within the training samples. Specifically, Yao et al. [2022] sets the standard deviation of the Gaussian noise as a fixed 30% of the range of the label space corresponding to the dataset. In contrast, our noise injection method introduces an element of stochasticity, allowing for variable levels of deviation for each unique label.

To achieve this variability, we employ uniform sampling from the minimum and maximum values specific to each label's domain. For instance, in the context of an age prediction task, we assume minimum and maximum values of 0 and 100, respectively. However, in cases where the label domain lacks clarity (*e.g.*, for a variable like 'price'), we utilize the minimum and maximum label values provided by the dataset itself.

It is important to highlight that baselines with known noise rates, such as CNLCU-S/H, Sigua, and Selfie, are incapable of dealing with Gaussian noise. Given that these baselines employ a heuristic approach to control selection rates through $(1 - \text{noise rate})$, they prove ineffective when exposed to Gaussian noise, as it introduces noise to all samples, thereby resulting in a nearly 100% noise rate. Hence, we create a *soft noise rate* to be used by them for selection. This is done by calculating an updated noise rate, assuming that the Gaussian noise injected samples that fall within an acceptable variance of the original ground-truth label are clean (the acceptable variance is set to equal the label length/size of a single fragment).

F.5 Computation Resource

For implementation, we use Python 3.9 and PyTorch 1.13.1. All experiments are conducted using NVIDIA Quadro 6000 24GB RAM GPUs. The required computation time for experiments differs depending on the dataset. Below, we report the computation time for each dataset.

AFAD-B. On the AFAD-B dataset, the average computation time of ConFrag and Co-ConFrag are approximately 2.5 GPU hours and 5 GPU hours, respectively. The computation time of baselines ranges from 1.25 GPU hours to 4 GPU hours. About 650 GPU hours were required to produce the AFAD-B part of Tab. 1.

IMDB-Clean-B. On the IMDB-Clean-B dataset, the average computation time of ConFrag and Co-ConFrag are approximately 3.5 GPU hours and 5.5 GPU hours, respectively. The computation time of baselines ranges from 2 GPU hours to 5 GPU hours. About 970 GPU hours were required to produce the IMDB-Clean-B part of Tab. 1.

SHIFT15M-B. On the SHIFT15M-B dataset, the average computation time of ConFrag and Co-ConFrag are approximately 1 GPU hour and 1.2 GPU hours, respectively. The computation time of baselines ranges from 6 GPU minutes to 1 GPU hour. About 100 GPU hours were required to produce the SHIFT15M-B part of Tab. 1.

MSD-B. On the MSD-B dataset, the average computation time of ConFrag and Co-ConFrag are approximately 4 GPU minutes and 5 GPU minutes, respectively. The computation time of baselines ranges from 1 GPU minute to 4 GPU minutes. About 9 GPU hours were required to produce the MSD-B part of Tab. 1.

Additionally, further computation was required for analysis and experiments in § 4.3 and Appendix G.

G Extended Results & Analysis

We conduct supplementary experiments and analyses of UTKFace dataset, parameter sizes, fragment numbers (F), other hyperparameters (K, J), fragment pairing, and the impact of closed-set and open-set noise. Furthermore, we present ablation analyses, comparisons with discretized baselines, baseline performance evaluations considering Selection rate and ERR, variance assessments, and the obtained MAE results.

G.1 UTKFace Results

Table 8 presents a comparison of MRAE values between various baseline methods and our proposed approach on the balanced UTKFace dataset [Zhifei et al., 2017], UTKFace-B. We experiment under four different symmetric noise conditions of symmteric 20%, 40%, 60% and 80% noise rates. Both ConFrag and Co-ConFrag demonstrate superior performance across all experiments when compared to the fourteen baseline methods.

Table 8: Comparison of MRAE (%) on UTKFace-B datasets with symmetric noise.

	UTKFace-B			
	symmetric			
	20	40	60	80
Vanilla	37.59	53.96	82.88	115.49
AUX	25.53	48.01	84.78	118.31
BMM	49.21	76.25	88.87	139.01
CDR	32.87	50.56	83.41	121.36
C-Mixup	17.76	34.00	74.29	117.68
CNLCU-H	5.75	20.75	43.44	121.55
CNLCU-S	20.29	36.69	43.44	121.55
D2L	38.28	51.22	99.19	122.50
DY-S	22.73	31.07	58.05	113.21
RDI	43.49	49.44	75.67	122.40
Selfie	30.47	44.62	99.67	130.97
Sigua	10.86	19.58	52.37	128.06
SPR	28.05	48.07	85.18	120.58
Superloss	9.12	22.10	55.78	115.78
ConFrag	6.28	14.30	34.09	83.03
Co-ConFrag	-2.22	8.88	25.46	74.78

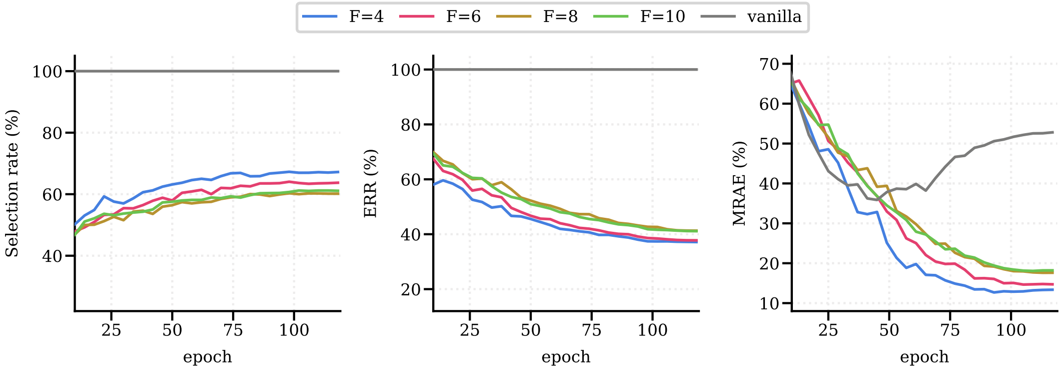


Figure 10: **Fragment number analysis** compares the Selection rate, ERR and MRAE on IMDB-Clean-B with symmetric 40% noise.

G.2 Fragment Number Analysis

In Fig. 10 and 11, we undertake an examination of various fragment numbers within the context of symmetric 40% noise, using the IMDB-Clean-B and SHIFT15M-B datasets as benchmarks. Our evaluation criteria encompass the Selection rate, Error Residual Rate (ERR), and Mean Relative Absolute Error (MRAE). The number of fragments is chosen from $F \in [4, 6, 8, 10]$. To address scenarios with a smaller fragment number, we examine cases where $F = 1$ or 2. When $F = 2$, a fragment f that satisfies self-agreement (Eq. 3) does not meet the criteria for neighbor-agreement (α_f^{ngb} in Eq. 4), as the agreement relies on comparing the distribution of fragment f and its contrasting pair f^+. Consequently, the unified neighborhood agreement (Eq. 4) consistently yields a value of 0. On the other hand, defining a contrasting pair is not feasible when $F = 1$. Instead, we present a plot of the vanilla baseline to illustrate the case when $F = 1$ without utilizing ConFrag.

The results reveal that the MRAE of the vanilla model initially decreases during the early epochs as it learns patterns from clean samples. However, as the model starts to memorize noisy samples, the MRAE degrades. In contrast, ConFrag consistently mitigates the impact of noisy samples across all plots ($F \in [4, 6, 8, 10]$) when compared to the vanilla baseline.

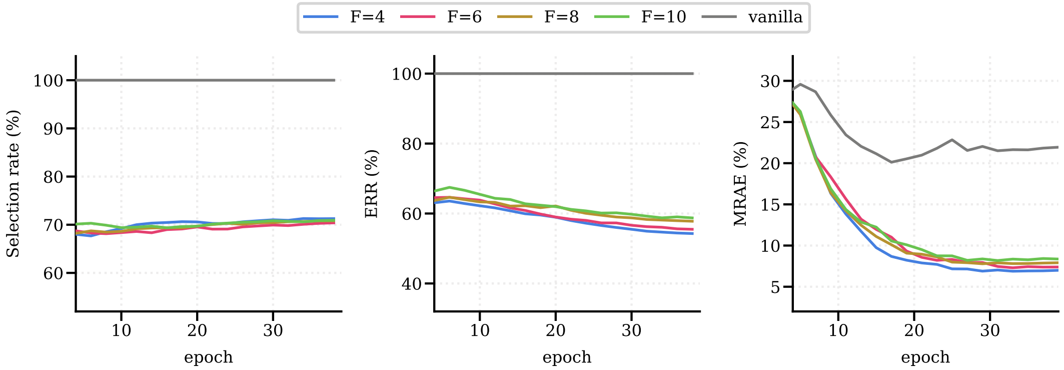


Figure 11: **Fragment number analysis** compares the Selection rate, ERR and MRAE on SHIFT15M-B with symmetric 40% noise.

We also observe a declining trend in performance as the number of fragments increases in the case of IMDB-Clean-B. In contrast, SHIFT15M-B exhibits relatively stable performance across different fragment numbers. This decrease in performance with an increased number of fragments is likely attributed to a finer division of the training data among feature extractors, ultimately leading to overfitting and reduced generalization capabilities.

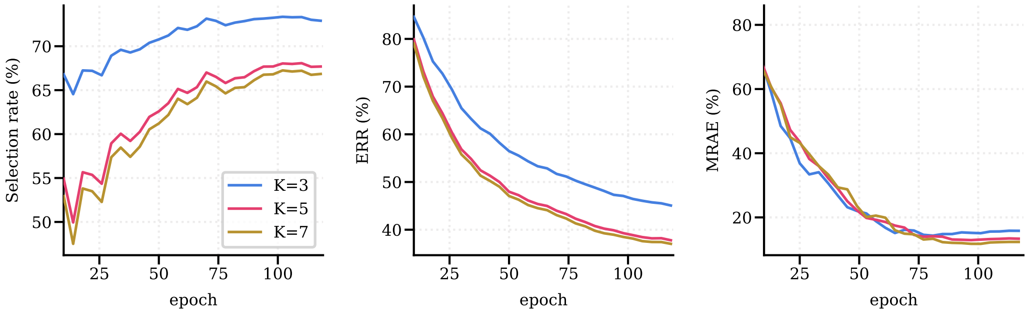


Figure 12: **Hyperparameter K analysis** compares the Selection rate, ERR and MRAE on IMDB-Clean-B with symmetric 40% noise.

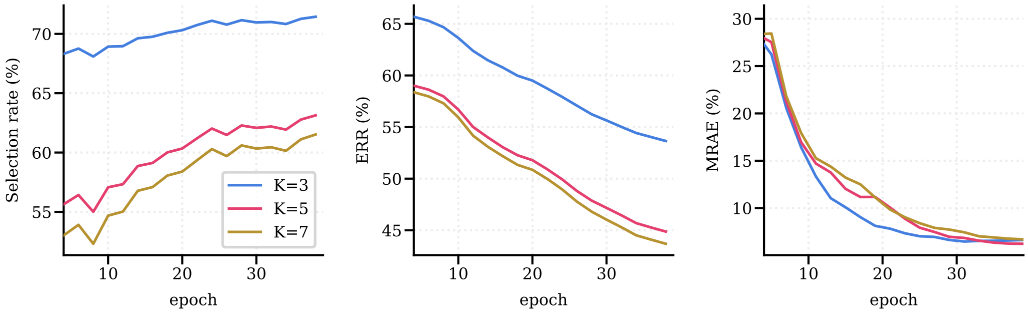
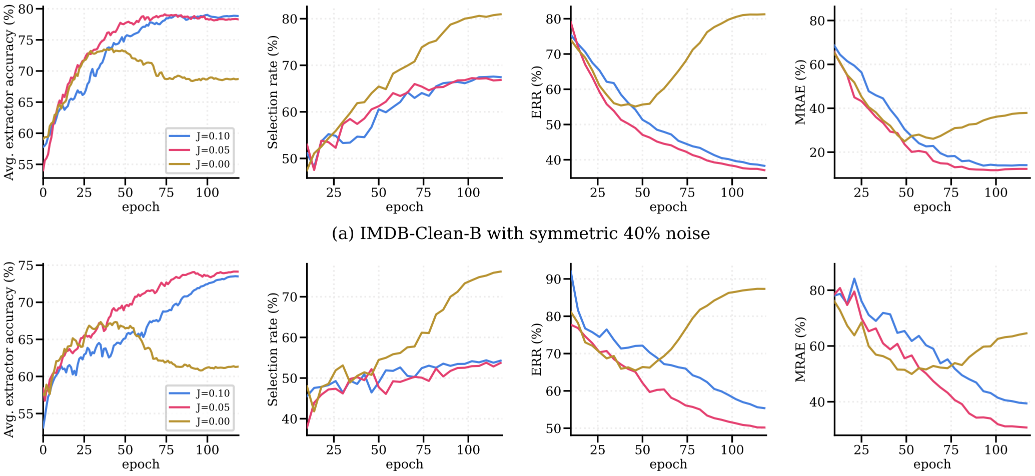


Figure 13: **Hyperparameter K analysis** compares the Selection rate, ERR and MRAE on SHIFT15M-B with symmetric 40% noise.

G.3 Hyperparameter Analysis

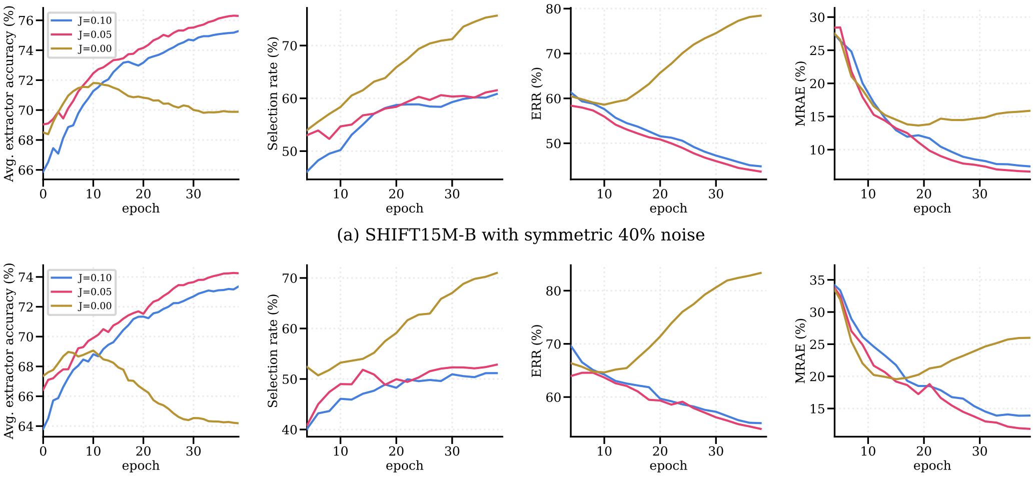
The hyperparameter K is used for K-nearest neighbor classification when assessing self/neighbor agreement from a representational perspective. As shown in Fig. 12, 13, with an increase in the value



(a) IMDB-Clean-B with symmetric 40% noise

(b) IMDB-Clean-B with symmetric 60% noise

Figure 14: **Hyperparameter J analysis** compares the average accuracy of feature extractors, the Selection rate, ERR and MRAE on IMDB-Clean-B with symmetric 40%, 60% noise.



(a) SHIFT15M-B with symmetric 40% noise

(b) SHIFT15M-B with symmetric 60% noise

Figure 15: **Hyperparameter J analysis** compares the average accuracy of feature extractors, the Selection rate, ERR and MRAE on SHIFT15M-B with symmetric 40%, 60% noise.

of K, the criteria for agreement become more stringent. Consequently, as the value of K increases, a greater number of confident samples are selected, resulting in a reduction in the Selection rate and ERR.

The hyperparameter J controls the buffer range for jittering, which, in turn, determines the level of regularization applied via neighborhood jittering. Increasing the value of J results in stronger regularization, effectively preventing overfitting. However, excessive regularization, as observed when $J = 0.10$, may result in adverse effects during training. Specifically, in Fig. 14(a), the feature extractors exhibit similar convergence patterns when $J = 0.05$ or $J = 0.10$. Consequently, comparable performance is observed in Selection Rate and MRAE. Yet, in Fig. 14(b), the ERR of $J = 0.05$ is smaller than that of $J = 0.10$, leading to improved MRAE performance for $J = 0.05$. Similar effects are observed in the SHIFT15M dataset, as depicted in Fig. 15 (SHIFT15M-B).

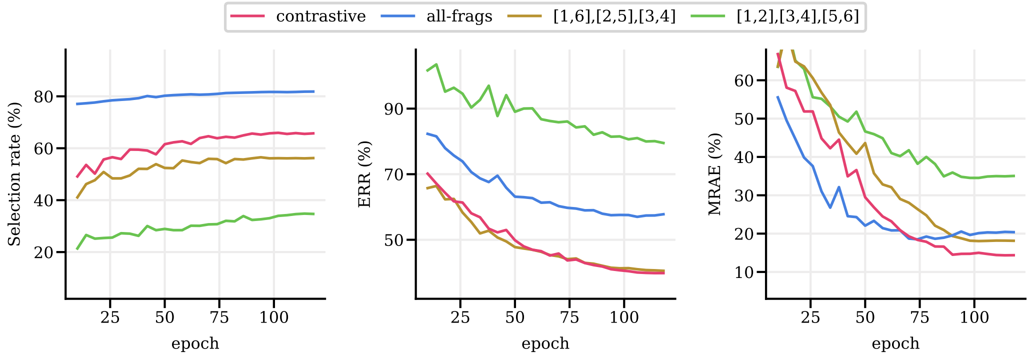


Figure 16: **Fragment pairing analysis** compares contrastive pairings ($[1, 4]$, $[2, 5]$, $[3, 6]$), all-fragments ($[1, 2, 3, 4, 5, 6]$), and alternative pairing methods ($[1, 2]$, $[3, 4]$, $[5, 6]$ and $[1, 6]$, $[2, 5]$, $[3, 4]$) on IMDB-Clean-B with 40% symmetric noise when $F = 6$. For feature extractor, all-fragments use a ResNet-34, while other pairing methods use ResNet-18 backbones.

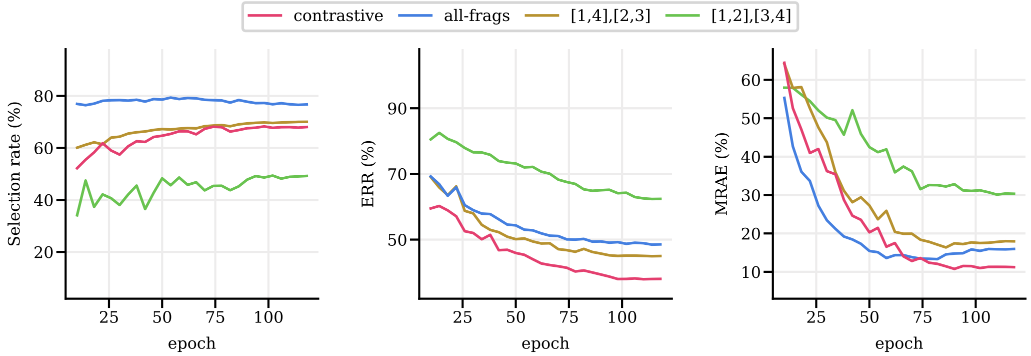


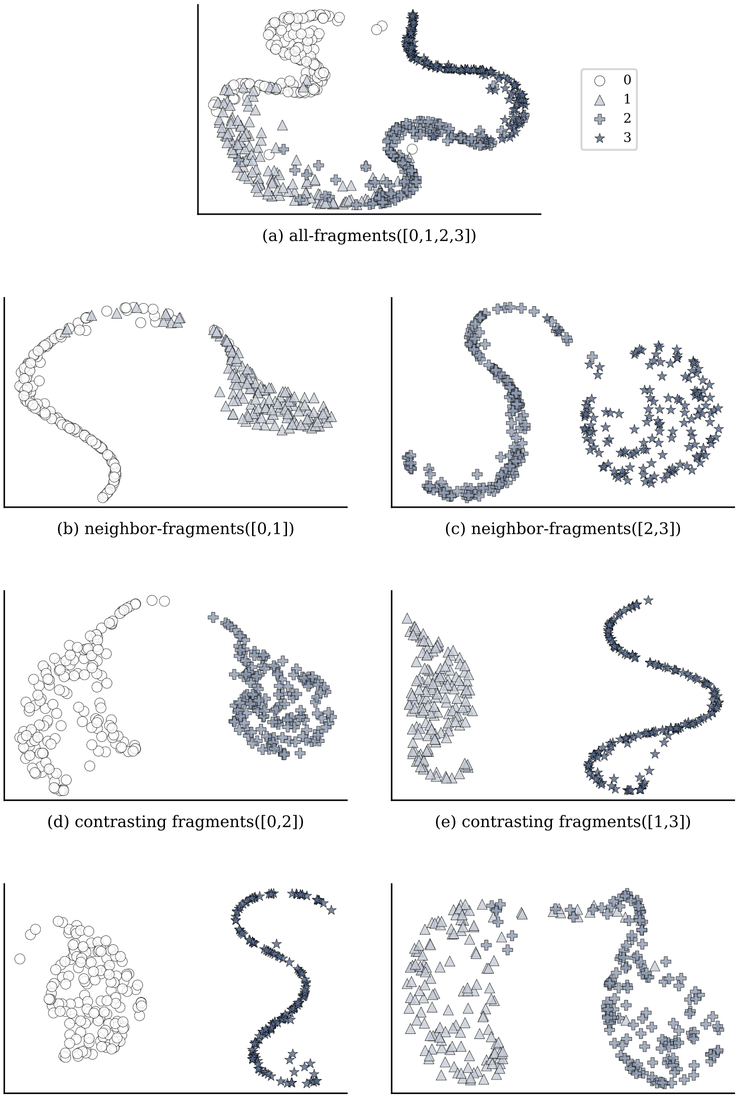
Figure 17: **Fragment pairing analysis** compares contrastive pairings ($[1, 3]$, $[2, 4]$), all-fragments ($[1, 2, 3, 4]$), and alternative pairing methods ($[1, 2]$, $[3, 4]$ and $[1, 4]$, $[2, 3]$) on IMDB-Clean-B with 40% symmetric noise when $F = 4$. For feature extractor, all-fragments use a ResNet-34, while other pairing methods use ResNet-18 backbones.

G.4 Fragment Pairing Analysis

In Fig. 1(c), we offer deeper insights into our approach by comparing contrastive fragment pairing ($[1, 4]$, $[2, 5]$, $[3, 6]$) against all-fragments ($[1, 2, 3, 4, 5, 6]$). In Fig. 6(a), we show the importance of contrastive fragment pairing by comparing contrastive fragment pairing to alternative pairings. In Fig. 16–17, we present the extended results with Selection rate, ERR, and MRAE alongside other pairing methods.

The experiments involve training the feature extractors using either contrastive fragment pairing, all-fragments, or alternative pairings. Notably, a single feature extractor is employed for all fragments, whereas the fragment pairing (contrastive or alternative) uses a smaller feature extractor for each individual pair. Subsequently, sample selection is executed in accordance with the Mixture of Neighboring Fragments approach (§ 2.3).

In an optimal selection algorithm, the Selection rate should approach $100 -$ noise rate$(\%)$, with ERR and MRAE minimized. Across all evaluation metrics, the contrastive fragment pairing demonstrates superior performance compared to other methods. It is important to highlight that performance is poorest when the pairing is least distinguishable ($[1, 2]$, $[3, 4]$, $[5, 6]$ when $F = 6$, $[1, 2]$, $[3, 4]$ when $F = 4$) and moderate when the pairing is partially distinguishable ($[1, 6]$, $[2, 5]$, $[3, 4]$ when $F = 6$, $[1, 4]$, $[2, 3]$ when $F = 4$).



(a) all-fragments([0,1,2,3])

(b) neighbor-fragments([0,1])

(c) neighbor-fragments([2,3])

(d) contrasting fragments([0,2])

(e) contrasting fragments([1,3])

(f) partial contrasting fragments([0,3])

(g) partial contrasting fragments([1,2])

Figure 18: **Detailed Representation Depiction**. A detailed comparison of the effect of fragment pairings via t-SNE visualization of the penultimate features from the feature extractors. The experiments are based on IMDB-Clean-B.

Furthermore, in Fig. 18, we utilize t-SNE to compare the feature extractors trained using contrastive pairing, alternative pairings, and all-fragments. The visual comparison validates that representations trained with contrastive pairs exhibit significantly more distinguishable features.

G.5 Closed-Set versus Open-Set Noise

To explore the impact of closed-set and open-set noisy samples, as depicted in Fig. 1(b) in the main manuscript, we conducted an analysis of Selection rate, ERR, and MRAE performance while gradually introducing closed-set and open-set noisy samples into the IMDB-Clean-B dataset. Our study employs the IMDB-Clean-B dataset, comprising a fixed set of clean samples that represent 40% of the total dataset, alongside varying amounts of noisy samples. These noisy samples are classified into two distinct categories: closed-set and open-set noise [Wei et al., 2021, Wan et al., 2024]. For example, consider an example with 4 fragments, whose contrastive fragment pairs are $\{(1,3),(2,4)\}$. When training a feature extractor on binary classification between fragment 1 and 3, noisy sample

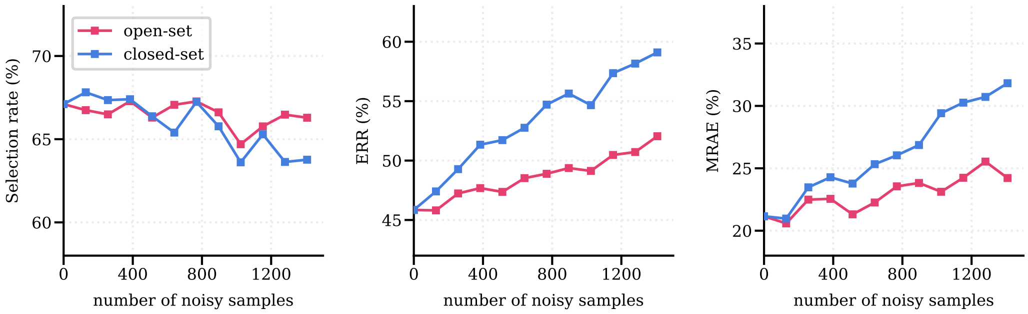


Figure 19: **Closed-set/open-set noise analysis** displays the selection, ERR and MRAE when closed-set or open-set noisy samples are injected into the clean dataset. The experiments are based on IMDB-Clean-B.

Table 9: Difference of selection rate and ERR between the samples at the boundary and center of fragments

Selection rate				
	20%	**40%**	**60%**	**80%**
boundary	79.55%	65.31%	55.98%	65.96%
center	80.18%	66.86%	54.64%	61.16%
difference	0.63%	1.55%	1.34%	4.80%
ERR				
boundary	31.90%	39.01%	55.44%	82.26%
center	30.16%	36.63%	50.42%	82.91%
difference	1.74%	2.38%	5.02%	0.65%

whose ground truth fragment id is either 2 or 4 but mislabeled as fragment id of either 1 or 3 is an open-set noisy sample. On the other hand, a noisy sample whose ground truth fragment id is 1 but mislabeled as 3 (and vice versa) is a closed-set noisy sample.

Fig. 19 demonstrates that closed-set noisy samples have a considerably more adverse impact on ERR and MRAE compared to open-set noisy samples. Our contrastive fragment pair-based learning approach is advantageous in this regard, as it introduces open-set noisy samples in lieu of many closed-set noisy samples, thereby facilitating learning with reduced interference.

G.6 Analysis of Samples on the Bounday versus Center of Fragments

Table 9 presents a comparative analysis of the selection rate and error reduction rate (ERR) between samples located at the boundary and the center of fragments across eight experimental configurations. The results indicate an average difference of 2.29% in selection rates and 2.43% in ERR between the two groups. These findings substantiate the robustness of ConFrag's sample selection process, demonstrating consistent performance irrespective of the sample's positional location within the fragment.

G.7 Ablation & Combination Analysis

In Table 10, we present a comprehensive study comparing the performance of Cross-Entropy (CE) and Symmetric Cross Entropy (SCE) [Wang et al., 2019] losses in various ablation and combination experiments conducted on the IMDB-Clean-B dataset [Yiming et al., 2021], considering scenarios with 40% symmetric noise and two variations of Gaussian random noise, each having a maximum standard deviation of 30 and 50.

Firstly, we illustrate the impact of jittering regularization through ablation on each of the losses. Notably, jittering regularization emerges as a crucial component of ConFrag's performance, preventing the model from overfitting to the noisy labels.

Table 10: **Ablation and Combination Analysis.** The values are mean relative absolute error to the noise-free trained model on the IMDB-Clean-B [Yiming et al., 2021] dataset, and lower values indicate better performances. The results are the mean of three random seed experiments.

ablation and combinations				IMDB-Clean-B		
				symmetric	Gaussian	
feat. ext. loss	backbone	jitter	Co-teaching	40	30	50
CE	ResNet-18			18.90	21.77	39.78
CE	ResNet-18	✓		12.64	15.70	33.36
CE	ResNet-34	✓		13.44	16.06	31.00
CE	ResNet-18	✓	✓	9.45	14.87	35.88
SCE	ResNet-18			18.37	20.80	38.10
SCE	ResNet-18	✓		16.84	20.07	38.18
SCE	ResNet-34	✓		14.97	18.95	36.12
SCE	ResNet-18	✓	✓	13.19	18.32	41.02

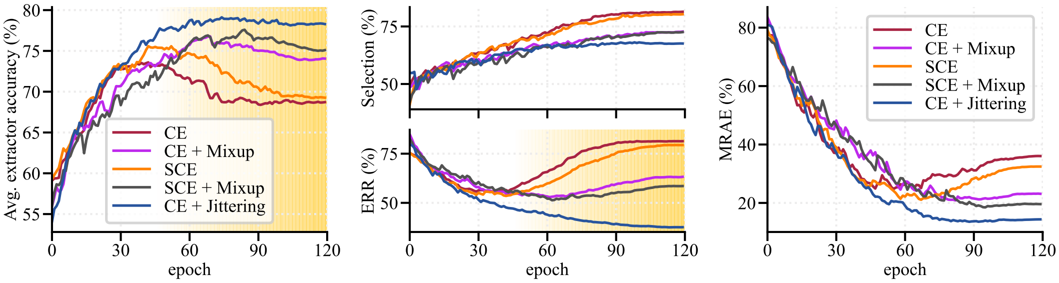
Table 11: **Discretized Baseline Analysis.** Mean Relative Absolute Error to the noise-free model of discretized versions of strongly performing models on the IMDB-Clean-B [Yiming et al., 2021] dataset. Lower is better.

	IMDB-Clean-B		
	symmetric	Gaussian	
noise rate (%)	40	30	50
CNLCU-S-D [Xia et al., 2022]	55.71	64.71	79.59
CNLCU-S-D + mixup [Xia et al., 2022]	55.14	67.17	81.32
CNLCU-H-D [Xia et al., 2022]	37.76	51.36	76.40
CNLCU-H-D + mixup [Xia et al., 2022]	65.32	67.31	84.22
Sigua-D [Han et al., 2020]	56.17	61.67	66.08
Sigua-D + mixup [Han et al., 2020]	33.55	29.33	49.44
BMM-D [Arazo et al., 2019]	33.86	30.27	50.05
MD-DYR-SH-D [Arazo et al., 2019]	33.89	31.18	51.23
CRUST-D [Mirzasoleiman et al., 2020]	33.86	30.27	50.47
CRUST-D + mixup [Mirzasoleiman et al., 2020]	32.33	30.50	50.27
Selfie-D [Song et al., 2019]	31.50	24.86	47.46
Selfie-D + mixup [Song et al., 2019]	35.33	28.02	46.42
Co-Selfie-D [Song et al., 2019]	30.20	26.36	49.61
Co-Selfie-D + mixup [Song et al., 2019]	33.18	28.28	52.20
ConFrag (Ours)	12.64	15.70	33.36
Co-ConFrag (Ours)	9.45	14.87	35.88

The next ablation experiment entails replacing the ResNet-18 architecture of the feature extractors with ResNet-34. The performance is enhanced when trained with SCE but decreases when trained with just CE. This suggests that ConFrag could potentially benefit from a more powerful architecture, but it is not a necessity.

A significant advantage of ConFrag lies in its compatibility with other approaches. We showcase its performance when combined with an additional technique: Co-teaching [Han et al., 2018], which is also employed by CNLCU and Co-Selfie in our baseline. Co-teaching involves training the regression model while heuristically assuming that 25% of the original noise still exists in the data (*e.g.*, 40% original noise implies an assumption of 10% noise during Co-teaching regression). Empirical observations reveal that Co-teaching consistently provides significant benefits.

Upon comparing CE and SCE for feature extractor training loss, we observe that CE, when combined with jitter regularization, synergizes better to exhibit much stronger performance compared to SCE.



(a) Feature extractor accuracy (b) Selection rate / ERR (c) Regression performance

Figure 20: **Comparison of regularization methods**. Compared to other regularization methods, neighborhood jittering demonstrates superior performance in (a) feature extractor test accuracy, (b) ERR, and (c) performance in regression. The analysis is conducted on IMDB-Clean-B with symmetric 40% noise.

G.8 Discretized Baselines

In Table 11, we present a discretized version of several strong baselines, including Sigua [Han et al., 2020], CNLCU [Xia et al., 2022], BMM [Arazo et al., 2019], Selfie/Co-Selfie [Song et al., 2019], MD-DYR-SH [Arazo et al., 2019], and CRUST [Mirzasoleiman et al., 2020].

The discretization process aligns with our fragmentation approach used for ConFrag. We obtain selected samples at the end of every epoch to independently train the regression model. Additionally, we report performance with mixup [Zhang et al., 2018], a technique that proves beneficial for some baselines like Sigua [Han et al., 2020].

Notably, most baselines exhibit a deterioration in performance following discretization. However, Selfie/Co-Selfie [Song et al., 2019] stands out as the exception, showing an improvement in performance after discretization. Interestingly, Sigua is the sole method that benefits from mixup [Zhang et al., 2018] training.

G.9 Comparison of Neighborhood Jittering and Other Regularization Methods

In Fig. 20, we compare neighborhood jittering with other regularization methods that can be applied to classification-based feature extractors (SCE with weight decay [Wang et al., 2019], mixup [Zhang et al., 2018], and their combinations). In conclusion, neighborhood jittering exhibits the strongest performance in feature extractor test accuracy, ERR, and MRAE, among other regularization methods. It is observed that ERR and MRAE improve in line with the performance of the feature extractor.

G.10 Extended Selection Rate/ERR/MRAE Comparison and Analysis

In addition to presenting the Selection rate, ERR and MRAE for symmetric 40%, Gaussian 30, and Gaussian 50 noise experiments on the IMDB-Clean-B dataset in the main manuscript, we have included results for all noise types, along with additional baselines (CNLCU-H, Sigua, BMM, DY-S, AUX, Selfie, Coselfie), in both Fig. 21 and Fig. 22.

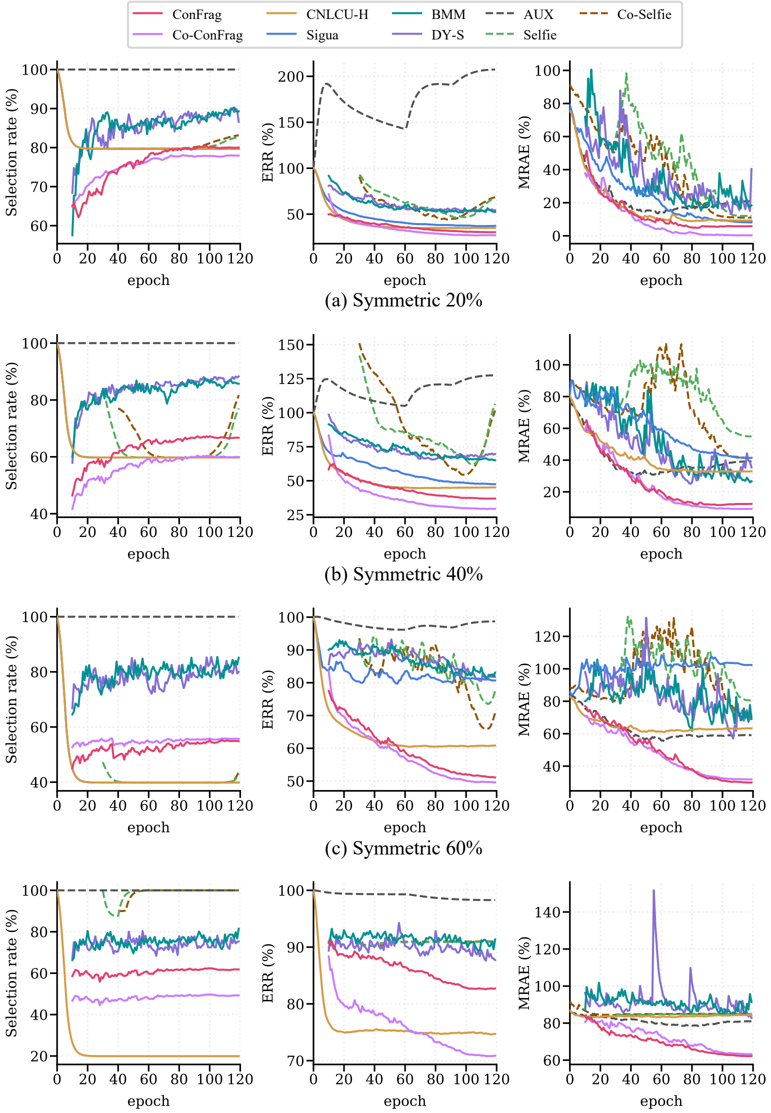
As mentioned in § 4.2, the ideal scenario for selection and refurbishment methods involves achieving a high selection rate while maintaining a low ERR, resulting in a reduced MRAE. We examine the relationship between the selection rate, ERR, and MRAE based on Fig. 21(b). As training progresses, ConFrag and other selection methods (CNLCU-H, Sigua, BMM, DY-S) approach the ideal condition, resulting in an improving trend in MRAE. ConFrag, in particular, comes closest to the ideal scenario, resulting in superior MRAE performance.

The most unfavorable scenario arises when there is a low selection rate coupled with a high ERR. Selfie exemplifies the scenario in Fig. 21(b), which is connected to a relatively worse MRAE.

The scenarios of the low selection rates with low ERR and the high selection rates with high ERR can be further examined using CNLCU-H and BMM. CNLCU-H demonstrates superior selection quality in terms of ERR, while BMM exhibits a higher quantity in the selection rate. This quality/quantity trade-off is linked to the observation that CNLCU-H and BMM show similar MRAE performance in Fig. 21(b). Additionally, Fig. 22(a) reveals that the selection rate gap widens, while the ERR gap narrows when compared to Fig. 21(b). This is associated with BMM outperforming CNLCU-H in terms of the MRAE.

It's important to note that, rather than employing the selection rate and ERR as indicators for MRAE, these metrics offer valuable insights when assessing selected or refurbished samples directly independent of any potential regularizing effects introduced by the underlying regression model.

In addition, upon a detailed analysis of the figures, it becomes evident that Co-ConFrag consistently achieves the lowest ERR across a wide range of noise types. Notably, it maintains a Selection rate of above 40% while maintaining low ERR even in the presence of severe noise conditions, which leads to outstanding MRAE performance.



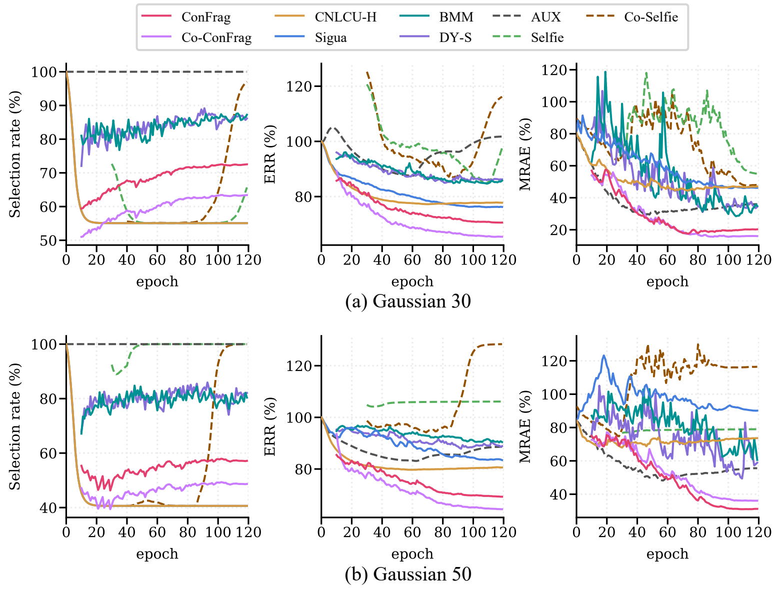
(a) Symmetric 20%

(b) Symmetric 40%

(c) Symmetric 60%

(d) Symmetric 80%

Figure 21: **Selection, ERR and MRAE comparison** of ConFrag, Co-ConFrag and filtering/refurbishment baselines on IMDB-Clean-B with symmetric 20%(a), 40%(b), 60%(c) and 80%(d) noise, repectively.



(a) Gaussian 30

(b) Gaussian 50

Figure 22: **Selection, ERR and MRAE comparison** of ConFrag, Co-ConFrag and filtering/refurbishment baselines on IMDB-Clean-B with Gaussian 30(a) and Gaussian 50(b) noise, repectively.

G.11 Variance Across Random Seeds

In Fig. 23, we plot the variance of three unique random seed experiments on all six noise types (symmetric 20%/40%/60%/80%, Gaussian 30/50) on the IMDB-Clean-B dataset. To declutter the graph, we compare it against the top two best-performing baselines under each noise type.

Tab. 12–13 show the main experimental results of Tab. 1 with standard deviation.

G.12 Standard Mean Absolute Error

In Tables 14–15, we report the standard mean absolute error (along with standard deviation) within the respective label ranges for each dataset.

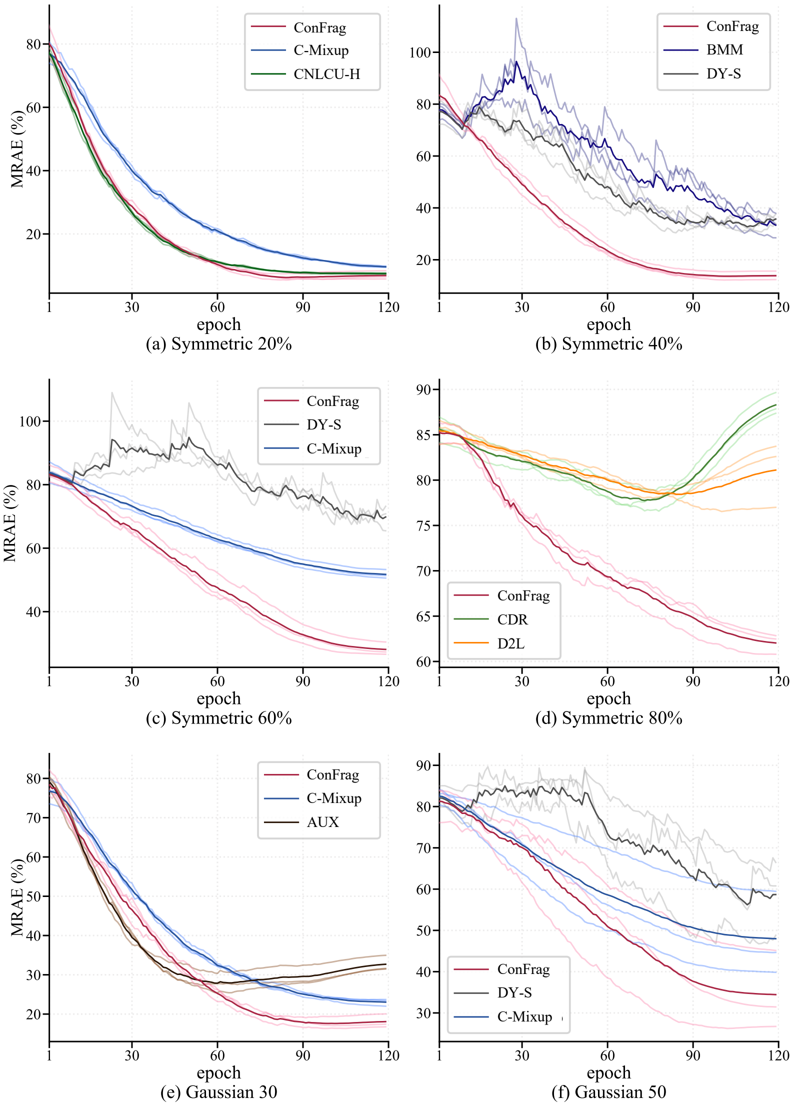


Figure 23: **Variance Analysis** of three unique random seed experiments on IMDB-Clean-B. The top two best-performing baselines under each noise type are reported.

Table 12: **Mean Relative Absolute Error (%)** and its standard deviation to the noise-free trained model on the AFAD-B, IMDB-Clean-B and IMDB-WIKI-B datasets. Lower is better. A negative value indicates it performs even better than the noise-free model. The results are the mean of three random seed experiments. Number in parenthesis indicates standard deviation. The best and the second best methods are respectively marked in red and blue. CNLCU-S/H, Co-Selfie, and Co-ConFrag use dual networks to teach each other as done in Han et al. [2018].

	AFAD-B						IMDB-Clean-B						IMDB-WIKI-B
	symmetric				Gaussian		symmetric				Gaussian		real noise
noise rate	20	40	60	80	30	50	20	40	60	80	30	50	-
Vanilla	9.37 (0.72)	20.27 (0.93)	30.65 (1.15)	43.09 (45.96)	28.77 (1.57)	39.03 (4.32)	16.18 (1.60)	32.05 (0.20)	53.13 (0.93)	76.35 (1.29)	26.89 (2.45)	50.28 (9.07)	00.00 (00.00)
CNLCU-S	10.98 (0.42)	20.44 (3.60)	32.44 (0.18)	41.99 (46.23)	30.60 (1.11)	40.66 (5.33)	51.40 (2.03)	66.62 (2.74)	82.83 (2.82)	85.65 (0.86)	83.39 (10.54)	82.10 (4.38)	21.54 (1.28)
CNLCU-H	4.63 (0.77)	16.32 (1.51)	36.01 (3.39)	44.71 (28.95)	35.68 (3.08)	43.64 (2.79)	6.84 (0.64)	31.16 (1.29)	63.08 (2.01)	82.65 (1.65)	46.53 (5.60)	65.24 (7.16)	-2.93 (0.82)
Sigua	5.96 (1.43)	21.09 (2.15)	43.33 (2.12)	49.71 (53.69)	42.52 (3.47)	46.19 (3.85)	9.82 (0.54)	46.17 (9.52)	77.59 (2.01)	85.62 (0.94)	60.97 (19.19)	77.42 (2.07)	1.96 (1.65)
SPR	9.74 (0.53)	18.85 (1.27)	30.43 (0.47)	43.25 (43.74)	28.50 (1.24)	39.69 (6.72)	14.47 (1.06)	32.44 (0.49)	54.88 (1.76)	79.37 (1.30)	25.67 (1.12)	51.05 (10.31)	-0.93 (1.61)
BMM	5.60 (0.68)	15.00 (1.91)	39.15 (3.16)	46.41 (44.77)	30.96 (6.89)	44.00 (2.11)	8.85 (1.53)	21.54 (2.29)	55.57 (8.88)	80.40 (3.18)	24.33 (1.49)	57.21 (16.74)	17.88 (2.05)
DY-S	6.87 (2.22)	15.56 (3.26)	32.24 (5.14)	45.72 (33.46)	24.40 (0.68)	43.41 (4.87)	10.42 (0.96)	21.90 (1.58)	49.94 (0.26)	78.16 (0.77)	24.70 (2.23)	44.56 (10.04)	-3.41 (0.86)
C-Mixup	2.74 (0.74)	14.80 (0.16)	27.17 (0.77)	41.95 (38.53)	24.28 (2.29)	36.91 (7.88)	8.82 (0.25)	27.74 (0.46)	50.87 (1.28)	76.79 (0.86)	21.92 (0.96)	47.04 (10.33)	-5.26 (0.52)
RDI	10.64 (0.41)	21.80 (0.66)	39.32 (0.76)	47.07 (49.28)	37.33 (1.51)	44.41 (4.41)	16.35 (1.09)	29.33 (1.98)	55.91 (0.60)	79.92 (0.69)	25.69 (1.75)	51.35 (12.37)	1.06 (0.67)
CDR	10.26 (1.20)	18.71 (1.03)	32.27 (0.66)	43.38 (45.06)	29.74 (0.41)	39.21 (6.15)	17.47 (1.11)	32.19 (1.54)	54.75 (1.72)	75.45 (0.89)	28.46 (2.89)	51.73 (7.38)	-0.39 (1.28)
D2L	9.43 (0.41)	20.75 (2.51)	31.25 (0.29)	44.50 (45.37)	28.86 (1.01)	40.10 (5.73)	16.94 (1.34)	33.85 (2.21)	55.54 (0.96)	76.28 (1.00)	29.30 (3.73)	52.44 (9.20)	-0.66 (0.82)
AUX	6.15 (0.17)	19.01 (0.71)	31.16 (0.50)	42.83 (44.84)	28.28 (1.70)	39.05 (4.81)	12.58 (0.66)	28.82 (1.35)	52.33 (0.82)	76.75 (1.08)	23.27 (1.78)	49.42 (9.86)	-3.67 (0.72)
Selfie	16.91 (4.09)	25.02 (3.42)	44.18 (0.48)	47.78 (42.55)	46.02 (5.90)	50.73 (4.93)	27.43 (15.12)	53.74 (2.07)	79.38 (0.08)	84.00 (0.28)	60.68 (5.42)	78.03 (3.46)	14.00 (11.45)
Co-Selfie	14.61 (2.66)	22.95 (1.03)	39.79 (1.33)	47.72 (35.29)	41.05 (6.05)	53.00 (15.38)	23.52 (12.18)	50.07 (11.39)	67.42 (3.08)	84.25 (0.59)	52.44 (8.15)	74.73 (6.99)	-0.44 (5.19))
Superloss	7.36 (2.02)	18.24 (1.38)	29.78 (1.64)	44.26 (40.38)	27.59 (1.09)	42.96 (5.80)	23.38 (4.49)	45.41 (3.14)	67.11 (1.88)	80.85 (1.66)	53.88 (16.22)	63.33 (8.88)	-3.58 (1.46)
ConFrag	2.74 (0.95)	8.16 (0.43)	15.91 (0.39)	34.42 (22.14)	17.49 (1.12)	27.31 (5.31)	5.08 (0.36)	12.64 (1.94)	27.26 (2.25)	61.24 (1.42)	15.70 (1.43)	33.36 (10.14)	-3.06 (1.25)
Co-ConFrag	0.54 (0.77)	7.25 (0.59)	16.65 (0.14)	33.93 (18.48)	17.43 (0.99)	28.26 (2.45)	1.50 (0.71)	9.45 (0.62)	28.44 (3.09)	61.36 (3.14)	14.87 (0.23)	35.88 (11.44)	-8.86 (0.83)

Table 13: **Mean Relative Absolute Error (%)** and its standard deviation to the noise-free trained model on the SHIFT15M-B and MSD-B datasets. Lower is better. A negative value indicates it performs even better than the noise-free model. The results are the mean of three random seed experiments. Number in parenthesis indicates standard deviation. The best and the second best methods are respectively marked in red and blue. CNLCU-S/H, Co-Selfie, and Co-ConFrag use dual networks to teach each other as done in Han et al. [2018]. SPR [Wang et al., 2022] fails to run for SHIFT15M-B due to excessive memory consumption.

	SHIFT15M-B						MSD-B					
	symmetric				Gaussian		symmetric				Gaussian	
noise rate	20	40	60	80	30	50	20	40	60	80	30	50
Vanilla	9.11	17.96	27.02	36.34	6.54	15.16	8.23	18.43	31.67	45.85	6.96	15.74
	(0.56)	(1.50)	(0.96)	(0.08)	(1.11)	(0.90)	(0.18)	(2.47)	(3.51)	(0.36)	(0.65)	(3.03)
CNLCU-S	12.98	19.42	24.31	34.47	15.33	20.90	0.13	6.04	21.52	46.01	4.75	12.51
	(0.15)	(0.39)	(0.70)	(0.17)	(1.09)	(0.34)	(1.18)	(0.31)	(3.61)	(1.51)	(0.92)	(1.08)
CNLCU-H	6.26	12.84	20.04	36.03	8.88	15.65	0.27	4.98	10.32	29.83	5.11	9.22
	(0.43)	(0.55)	(0.42)	(0.80)	(0.36)	(0.19)	(0.19)	(0.94)	(1.46)	(1.33)	(0.31)	(0.39)
Sigua	6.94	14.09	26.08	37.03	10.32	17.44	1.29	7.19	17.35	50.87	6.80	12.38
	(0.13)	(0.17)	(0.29)	(2.84)	(0.77)	(0.70)	(0.21)	(0.64)	(3.30)	(3.33)	(1.44)	(0.10)
SPR	-	-	-	-	-	-	7.07	18.19	33.39	45.61	5.01	15.36
	(-)	(-)	(-)	(-)	(-)	(-)	(1.30)	(1.32)	(3.17)	(2.11)	(0.31)	(2.15)
BMM	6.96	12.42	18.64	26.79	7.58	13.13	3.32	10.30	23.40	43.56	5.29	11.85
	(0.52)	(1.16)	(0.78)	(1.04)	(1.24)	(0.55)	(0.64)	(1.85)	(2.31)	(1.95)	(0.63)	(0.77)
DY-S	7.11	11.94	18.85	29.04	6.90	13.50	3.39	8.06	18.65	35.24	4.77	9.83
	(0.25)	(0.74)	(0.10)	(1.48)	(0.97)	(0.95)	(1.07)	(1.19)	(3.50)	(1.83)	(1.28)	(1.33)
C-Mixup	9.47	16.15	24.08	34.17	5.88	14.51	3.75	13.13	26.73	40.90	2.96	10.97
	(0.23)	(0.83)	(0.32)	(0.40)	(1.03)	(0.98)	(0.83)	(2.34)	(2.16)	(2.07)	(0.17)	(0.38)
RDI	9.91	17.92	26.63	36.29	7.08	15.18	21.04	30.09	38.78	49.49	19.19	27.88
	(0.45)	(0.27)	(0.36)	(0.67)	(0.85)	(0.84)	(0.09)	(0.22)	(0.98)	(1.12)	(0.69)	(1.60)
CDR	9.52	17.78	26.97	35.97	7.14	15.17	7.83	17.86	32.83	45.91	6.73	16.92
	(1.17)	(0.83)	(0.36)	(0.73)	(1.01)	(0.72)	(0.91)	(3.23)	(2.36)	(0.74)	(0.49)	(1.55)
D2L	9.25	18.03	26.55	36.23	6.34	15.60	7.13	19.96	32.47	46.64	5.51	15.54
	(0.26)	(1.47)	(1.13)	(0.66)	(0.69)	(1.58)	(0.37)	(1.08)	(2.21)	(2.56)	(0.76)	(2.05)
AUX	7.74	16.95	26.61	36.47	4.92	14.40	6.12	18.18	31.09	45.70	5.21	15.45
	(0.33)	(1.03)	(0.30)	(0.50)	(1.11)	(0.94)	(0.88)	(1.55)	(3.07)	(1.43)	(0.28)	(1.78)
Selfie	4.84	10.22	22.28	38.15	5.51	11.58	1.43	8.40	20.24	45.87	14.37	24.13
	(0.77)	(0.71)	(2.82)	(0.42)	(0.97)	(0.45)	(0.24)	(1.30)	(4.61)	(2.88)	(3.28)	(3.41)
Co-Selfie	11.53	16.43	32.08	39.32	13.45	22.33	-0.38	4.41	8.32	35.47	6.78	13.15
	(0.84)	(0.62)	(0.64)	(0.54)	(0.74)	(0.85)	(0.12)	(0.68)	(1.40)	(0.57)	(1.70)	(1.60)
Superloss	5.44	12.26	23.23	35.24	5.60	13.28	-0.15	10.68	23.15	45.55	4.35	16.36
	(1.03)	(1.48)	(1.89)	(0.28)	(1.28)	(0.67)	(0.29)	(2.10)	(3.15)	(6.77)	(0.74)	(2.99)
ConFrag	2.46	6.18	10.68	19.04	3.66	8.09	0.57	4.94	11.22	23.41	2.39	6.49
	(0.42)	(0.45)	(0.65)	(0.63)	(0.37)	(0.05)	(0.43)	(0.34)	(1.38)	(2.00)	(0.84)	(1.90)
Co-ConFrag	0.85	5.52	10.80	18.83	3.03	8.70	-0.65	2.98	8.66	20.53	1.73	6.00
	(0.31)	(0.66)	(0.43)	(0.41)	(0.94)	(0.46)	(0.72)	(0.66)	(0.36)	(2.46)	(1.02)	(1.07)

Table 14: **Standard Mean Absolute Error** and its standard deviation to the noise-free trained model on the AFAD-B, IMDB-Clean-B and IMDB-WIKI-B datasets. Lower is better. The results are the mean of three random seed experiments. Number in parenthesis indicates standard deviation. The best and the second best methods are respectively marked in red and blue. CNLCU-S/H, Co-Selfie, and Co-ConFrag use dual networks to teach each other as done in Han et al. [2018].

	AFAD-B						IMDB-Clean-B						IMDB-WIKI-B
	symmetric				Gaussian		symmetric				Gaussian		real noise
noise rate	20	40	60	80	30	50	20	40	60	80	30	50	-
Vanilla	4.75	5.22	5.68	6.22	5.59	6.04	8.11	9.22	10.70	12.32	8.86	10.50	7.23
	(0.02)	(0.03)	(0.05)	(0.06)	(0.08)	(0.18)	(0.09)	(0.05)	(0.12)	(0.06)	(0.13)	(0.60)	(0.09)
CNLCU-S	4.82	5.23	5.75	6.17	5.67	6.11	10.57	11.64	12.77	12.97	12.81	12.72	8.78
	(0.03)	(0.14)	(0.01)	(0.09)	(0.07)	(0.22)	(0.15)	(0.20)	(0.15)	(0.04)	(0.78)	(0.25)	(0.16)
CNLCU-H	4.55	5.05	5.91	6.29	5.89	6.24	7.46	9.16	11.39	12.76	10.24	11.54	7.01
	(0.05)	(0.05)	(0.16)	(0.08)	(0.15)	(0.11)	(0.08)	(0.12)	(0.14)	(0.18)	(0.43)	(0.47)	(0.02)
Sigua	4.60	5.26	6.23	6.50	6.19	6.35	7.67	10.21	12.40	12.96	11.25	12.39	7.37
	(0.07)	(0.10)	(0.07)	(0.04)	(0.17)	(0.17)	(0.04)	(0.67)	(0.08)	(0.04)	(1.37)	(0.19)	(0.12)
SPR	4.77	5.16	5.67	6.22	5.58	6.07	8.00	9.25	10.82	12.53	8.78	10.55	7.16
	(0.04)	(0.04)	(0.03)	(0.06)	(0.07)	(0.28)	(0.09)	(0.02)	(0.08)	(0.12)	(0.05)	(0.70)	(0.03)
BMM	4.59	5.00	6.04	6.36	5.69	6.26	7.60	8.49	10.87	12.60	8.68	10.98	8.52
	(0.03)	(0.09)	(0.12)	(0.06)	(0.31)	(0.09)	(0.11)	(0.20)	(0.68)	(0.21)	(0.12)	(1.16)	(0.13)
DY-S	4.64	5.02	5.74	6.33	5.40	6.23	7.71	8.51	10.47	12.44	8.71	10.10	6.98
	(0.09)	(0.12)	(0.24)	(0.16)	(0.05)	(0.20)	(0.11)	(0.13)	(0.04)	(0.05)	(0.14)	(0.68)	(0.07)
C-Mixup	4.46	4.99	5.52	6.17	5.40	5.95	7.60	8.92	10.54	12.35	8.52	10.27	6.84
	(0.04)	(0.02)	(0.05)	(0.07)	(0.12)	(0.34)	(0.06)	(0.04)	(0.06)	(0.12)	(0.04)	(0.69)	(0.04)
RDI	4.81	5.29	6.05	6.39	5.97	6.27	8.13	9.03	10.89	12.57	8.78	10.57	7.30
	(0.02)	(0.04)	(0.02)	(0.02)	(0.09)	(0.18)	(0.04)	(0.19)	(0.04)	(0.11)	(0.10)	(0.84)	(0.04)
CDR	4.79	5.16	5.75	6.23	5.64	6.05	8.20	9.23	10.81	12.25	8.97	10.60	7.20
	(0.04)	(0.03)	(0.03)	(0.10)	(0.03)	(0.26)	(0.08)	(0.16)	(0.16)	(0.02)	(0.16)	(0.48)	(0.01)
D2L	4.75	5.24	5.70	6.28	5.60	6.09	8.17	9.35	10.86	12.31	9.03	10.65	7.18
	(0.03)	(0.10)	(0.03)	(0.11)	(0.03)	(0.24)	(0.05)	(0.11)	(0.09)	(0.01)	(0.21)	(0.62)	(0.12)
AUX	4.61	5.17	5.70	6.20	5.57	6.04	7.86	9.00	10.64	12.35	8.61	10.44	6.96
	(0.02)	(0.05)	(0.04)	(0.07)	(0.09)	(0.20)	(0.01)	(0.11)	(0.11)	(0.13)	(0.13)	(0.66)	(0.05)
Selfie	5.08	5.43	6.26	6.42	6.34	6.55	8.90	10.74	12.53	12.85	11.22	12.43	8.24
	(0.19)	(0.16)	(0.01)	(0.01)	(0.28)	(0.23)	(1.07)	(0.10)	(0.07)	(0.05)	(0.39)	(0.21)	(0.92)
Co-Selfie	4.98	5.34	6.07	6.42	6.13	6.65	8.63	10.48	11.69	12.87	10.65	12.20	7.20
	(0.13)	(0.06)	(0.08)	(0.08)	(0.28)	(0.68)	(0.88)	(0.84)	(0.19)	(0.03)	(0.61)	(0.45)	(0.46)
Superloss	4.66	5.14	5.64	6.27	5.54	6.21	8.62	10.16	11.67	12.63	10.75	11.41	6.97
	(0.07)	(0.05)	(0.05)	(0.05)	(0.05)	(0.25)	(0.27)	(0.25)	(0.19)	(0.15)	(1.14)	(0.57)	(0.05)
ConFrag	4.46	4.70	5.04	5.84	5.10	5.53	7.34	7.87	8.89	11.26	8.08	9.31	7.00
	(0.04)	(0.02)	(0.03)	(0.05)	(0.06)	(0.22)	(0.04)	(0.15)	(0.12)	(0.04)	(0.07)	(0.68)	(0.13)
Co-ConFrag	4.37	4.66	5.07	5.82	5.10	5.57	7.09	7.64	8.97	11.27	8.02	9.49	6.58
	(0.05)	(0.04)	(0.01)	(0.09)	(0.06)	(0.11)	(0.08)	(0.05)	(0.20)	(0.16)	(0.05)	(0.76)	(0.06)

Table 15: **Standard Mean Absolute Error** and its standard deviation to the noise-free trained model on the SHIFT15M-B and MSD-B datasets. Lower is better. The results are the mean of three random seed experiments. Number in parenthesis indicates standard deviation. The best and the second best methods are respectively marked in red and blue. CNLCU-S/H, Co-Selfie, and Co-ConFrag use dual networks to teach each other as done in Han et al. [2018]. SPR [Wang et al., 2022] fails to run for SHIFT15M-B due to excessive memory consumption.

	SHIFT15M-B						MSD-B					
	symmetric				Gaussian		symmetric				Gaussian	
noise rate	20	40	60	80	30	50	20	40	60	80	30	50
Vanilla	7.47	8.08	8.70	9.34	7.30	7.89	.5918	.6475	.7199	.7974	.5848	.6328
	(0.02)	(0.07)	(0.08)	(0.04)	(0.05)	(0.08)	(.0024)	(.0112)	(.0171)	(.0016)	(.0031)	(.0161)
CNLCU-S	7.74	8.18	8.51	9.21	7.90	8.28	.5475	.5798	.6644	.7983	.5727	.6151
	(0.03)	(0.04)	(0.01)	(0.05)	(0.04)	(0.02)	(.0068)	(.0010)	(.0176)	(.0052)	(.0033)	(.0035)
CNLCU-H	7.28	7.73	8.22	9.32	7.46	7.92	.5483	.5740	.6032	.7098	.5747	.5972
	(0.03)	(0.04)	(0.01)	(0.05)	(0.01)	(0.03)	(.0034)	(.0027)	(.0055)	(.0065)	(.0040)	(.0019)
Sigua	7.32	7.81	8.64	9.39	7.56	8.04	.5538	.5861	.6416	.8248	.5839	.6145
	(0.03)	(0.03)	(0.06)	(0.24)	(0.02)	(0.08)	(.0035)	(.0024)	(.0154)	(.0150)	(.0062)	(.0026)
SPR	-	-	-	-	-	-	.5854	.6462	.7293	.7961	.5741	.6308
	(-)	(-)	(-)	(-)	(-)	(-)	(.0059)	(.0048)	(.0147)	(.0113)	(.0028)	(.0119)
BMM	7.33	7.70	8.13	8.68	7.37	7.75	.5649	.6031	.6747	.7849	.5757	.6116
	(0.03)	(0.05)	(0.03)	(0.08)	(0.06)	(0.07)	(.0044)	(.0081)	(.0099)	(.0073)	(.0016)	(.0025)
DY-S	7.34	7.67	8.14	8.84	7.32	7.77	.5653	.5908	.6487	.7394	.5728	.6005
	(0.02)	(0.03)	(0.04)	(0.10)	(0.03)	(0.10)	(.0072)	(.0040)	(.0164)	(.0102)	(.0045)	(.0049)
C-Mixup	7.50	7.95	8.50	9.19	7.25	7.84	.5673	.6185	.6929	.7704	.5630	.6067
	(0.03)	(0.03)	(0.03)	(0.05)	(0.04)	(0.08)	(.0041)	(.0107)	(.0096)	(.0135)	(.0023)	(.0028)
RDI	7.53	8.08	8.67	9.33	7.33	7.89	.6618	.7113	.7588	.8174	.6517	.6992
	(0.02)	(0.02)	(0.05)	(0.03)	(0.02)	(0.08)	(.0030)	(.0040)	(.0050)	(.0039)	(.0065)	(.0092)
CDR	7.50	8.07	8.70	9.31	7.34	7.89	.5896	.6444	.7262	.7978	.5836	.6393
	(0.05)	(0.03)	(0.05)	(0.02)	(0.04)	(0.08)	(.0065)	(.0160)	(.0116)	(.0062)	(.0030)	(.0089)
D2L	7.48	8.08	8.67	9.33	7.28	7.92	.5857	.6559	.7243	.8018	.5769	.6317
	(0.03)	(0.07)	(0.07)	(0.01)	(0.01)	(0.13)	(.0021)	(.0049)	(.0116)	(.0122)	(.0030)	(.0106)
AUX	7.38	8.01	8.67	9.35	7.19	7.83	.5802	.6462	.7167	.7966	.5753	.6312
	(0.01)	(0.04)	(0.04)	(0.04)	(0.04)	(0.08)	(.0027)	(.0097)	(.0152)	(.0067)	(.0010)	(.0089)
Selfie	7.18	7.55	8.37	9.46	7.23	7.64	.5546	.5927	.6574	.7976	.6253	.6787
	(0.03)	(0.03)	(0.19)	(0.02)	(0.04)	(0.05)	(.0020)	(.0046)	(.0230)	(.0174)	(.0190)	(.0175)
Co-Selfie	7.64	7.97	9.05	9.54	7.77	8.38	.5447	.5709	.5923	.7407	.5839	.6187
	(0.03)	(0.01)	(0.01)	(0.01)	(0.02)	(0.08)	(.0026)	(.0019)	(.0088)	(.0023)	(.0105)	(.0084)
Superloss	7.22	7.69	8.44	9.26	7.23	7.76	.5460	.6052	.6733	.7959	.5706	.6362
	(0.06)	(0.07)	(0.09)	(0.06)	(0.06)	(0.06)	(.0036)	(.0104)	(.0153)	(.0405)	(.0065)	(.0140)
ConFrag	7.02	7.27	7.58	8.15	7.10	7.40	.5499	.5738	.6081	.6747	.5598	.5822
	(0.01)	(0.02)	(0.01)	(0.03)	(0.01)	(0.04)	(.0035)	(.0039)	(.0051)	(.0098)	(.0050)	(.0084)
Co-ConFrag	6.91	7.23	7.59	8.14	7.06	7.44	.5432	.5631	.5941	.6590	.5562	.5796
	(0.01)	(0.01)	(0.01)	(0.06)	(0.04)	(0.06)	(.0056)	(.0018)	(.0009)	(.0129)	(.0051)	(.0044)

Algorithm 1 Contrastive Fragmentation

```
**Input:** Train data $\mathcal{D} = \{\mathcal{X}, Y\}$, Fragment number $F$, KNN parameter $K$, Jitter $J$, Total epochs $N$

$\Theta = \{\theta_{0,0} \dots \theta_{i,j}\}$ {feature extractors}
$\Phi = RandomInit()$ {regression model}

$\mathcal{D}_{1 \dots F} = Fragmentation(\mathcal{D})$ {§ 2.1. 1}
$\mathcal{P} = ContrastivePairing(\mathcal{D}_{1 \dots F})$ {§ 2.1. 2∼4}
**for** $n$ **to** $N$ **do**
  # train feature extractors
  $\mathcal{P}^{jitter} = NeighborhoodJittering(\mathcal{D}, F, J)$ {neighborhood jittering (§ 2.4)}
  **for** $(\mathcal{D}_i^{jitter}, \mathcal{D}_j^{jitter})$ **in** $\mathcal{P}^{jitter}$ **do**
    $\mathcal{D}_{i,j}^{jitter} = \mathcal{D}_i^{jitter} \cup \mathcal{D}_j^{jitter}$
    train $p(f; \theta_{i,j}, \mathcal{D}_{i,j}^{jitter})$
  **end for**

  # initialize $\mathcal{S}, \mathcal{S}^p, \mathcal{S}^r$
  $\mathcal{S}, \mathcal{S}^p, \mathcal{S}^r = \{\}, \{\}, \{\}$ {selected samples}

  # obtain $\mathcal{S}^p, \mathcal{S}^r$
  **for** $(x, y)$ **in** $\mathcal{D}$ **do**
    **for** $f = 1$ **to** $F$ **do**
      calculate $\rho_f(y)$ {fragment prior (Eq. 2)}
      # use two types of classification for neighborhood agreement
      calculate $\alpha_f^p(x; \mathcal{D}_{1 \dots F}, \Theta)$ {predictive neighborhood agreement (Eq. 4)}
      calculate $\alpha_f^r(x; \mathcal{D}_{1 \dots F}, \Theta)$ {representational neighborhood agreement (Eq. 4)}
    **end for**
    $p^p(s|x, y, \mathcal{D}_{1 \dots F}; \Theta) = \sum_f^F \rho_f(y) \alpha_f^p(x; \mathcal{D}_{1 \dots F}, \Theta)$ {pred. sample probability (Eq. 1)}
    $p^r(s|x, y, \mathcal{D}_{1 \dots F}; \Theta) = \sum_f^F \rho_f(y) \alpha_f^r(x; \mathcal{D}_{1 \dots F}, \Theta)$ {repr. sample probability (Eq. 1)}
    sample $\{u^p, u^r\} \sim uniform(0, 1)$
    **if** $p^p(s|x, y, \mathcal{D}_{1 \dots F}; \Theta) > u^p$ **then**
      $\mathcal{S}^p = \mathcal{S}^p \cup (x, y)$
    **end if**
    **if** $p^r(s|x, y, \mathcal{D}_{1 \dots F}; \Theta) > u^r$ **then**
      $\mathcal{S}^r = \mathcal{S}^r \cup (x, y)$
    **end if**
  **end for**

  # union filtered samples $(\mathcal{S}^p, \mathcal{S}^r)$
  $\mathcal{S} = \mathcal{S}^p \cup \mathcal{S}^r$

  # train regression model
  $\Phi = TrainOneEpoch(S; \Phi)$

**end for**
```

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The contributions of the paper are outlined in the introduction and match the experimental results of the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of the work are discussed in Appendix B.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Detailed information required to reproduce the main experimental results are provided in § 4.1 and Appendix F.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide open access to code for reproducing the main experimental results.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setting is presented in § 4.1 with the full details provided in Appendix F.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Appendix G.11 contains main experimental results with standard deviation calculated over three random seeds, along with visualizations of the variance.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix F.5 provides the computation resource used and time of execution.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: All authors read and confirm that the research in the paper conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The broader societal impacts of the work is described in Appendix C.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper proposes a general machine learning method and thus the paper poses no such risk of misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper properly cites the original assets such as datasets, and Appendix F.1 provides the licenses of existing dataset.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The provided code includes documentation on training and dataset construction.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.