

RankSim: Ranking Similarity Regularization for Deep Imbalanced Regression

Yu Gong^{*12} Greg Mori¹² Frederick Tung²

Abstract

Data imbalance, in which a plurality of the data samples come from a small proportion of labels, poses a challenge in training deep neural networks. Unlike classification, in regression the labels are continuous, potentially boundless, and form a natural ordering. These distinct features of regression call for new techniques that leverage the additional information encoded in label-space relationships. This paper presents the RankSim (ranking similarity) regularizer for deep imbalanced regression, which encodes an inductive bias that samples that are closer in label space should also be closer in feature space. In contrast to recent distribution smoothing based approaches, RankSim captures both nearby and distant relationships: for a given data sample, RankSim encourages the sorted list of its neighbors in label space to match the sorted list of its neighbors in feature space. RankSim is complementary to conventional imbalanced learning techniques, including re-weighting, two-stage training, and distribution smoothing, and lifts the state-of-the-art performance on three imbalanced regression benchmarks: IMDB-WIKI-DIR, AgeDB-DIR, and STS-B-DIR.

1. Introduction

Skewed data distributions, in which a plurality of the data instances are from a small number of labels, widely exist in the real world and pose challenges to conventional machine learning techniques. Although deep learning has made much progress and shown its potential in many application domains, imbalanced datasets often present practical difficulties, causing models to exhibit undesirable bias towards the majority labels and under-representing the minority labels. Prior work has thoroughly studied how to

^{*}This work was done during an internship at Borealis AI.
¹Simon Fraser University ²Borealis AI. Correspondence to: Yu Gong <gongyug@sfu.ca>.

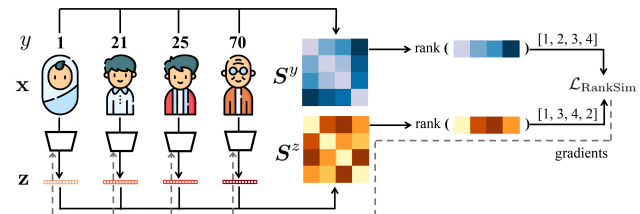


Figure 1. The RankSim (ranking similarity) regularizer for deep imbalanced regression introduces an inductive bias that items that are closer in label space should also be closer in feature space. Matrices S^y and S^z encode pairwise similarities in label space and feature space, respectively. For a given input sample, RankSim encourages the sorted list of its neighbors in label space to match the sorted list of its neighbors in feature space.

improve model training on imbalanced data, such as via cost-sensitive learning, representation learning, and decoupled learning. However, previous attempts have almost exclusively focused on the task of imbalanced classification. A very recent work (Yang et al., 2021) drew attention to the nuances of regression with imbalanced data and identified the task of deep imbalanced regression as a distinct problem.

Compared to classification, how to train deep regression models with imbalanced data is not yet as well understood. Unlike classification networks predicting discrete labels to model the categorical distributions, regression networks aim to predict labels with continuous values. The continuity in label space makes deep imbalanced regression different from deep imbalanced classification. On the one hand, the target values can be infinite and boundless, which makes many methods designed for deep imbalanced classification untenable. On the other hand, the continuity in label space can also play a positive role by providing extra information about data instance relationships.

In regression, the prediction targets form a natural ordering. For example, if we are interested in age estimation, we may order people from youngest to oldest; if we are performing home valuation, we may order properties from cheapest to most expensive. We can use this natural ordering of labels to regularize the representation learned by a neural network. For example, we can nudge the network in a direction such that the feature representation for the face of a 21-year old

is more similar to the representation for a 25-year old than for a 70-year-old. This intuition of preserving label-space relationships in the learned feature space is the motivation behind recent smoothing-based regularization approaches to address the imbalanced data problem in regression, such as label distribution smoothing (LDS) and feature distribution smoothing (FDS) (Yang et al., 2021). As an illustrative example of why preserving label-space relationships helps with imbalanced regression, when Yang et al. (2021) trained vanilla models on an imbalanced age estimation dataset, they observed that the learned features of 0-6 year-olds, which have very few samples, were highly similar to the learned features of 30 year-olds, which have a large number of samples. This proximity in feature space, despite the large difference in label space, is undesirable as it hinders generalization to unseen 0-6 year-olds.

Our proposed RankSim regularizer builds on this intuition and applies a stronger inductive bias. We provide an illustration in Fig. 1. Using age estimation as an example, suppose we have images of individuals whose ages are 1, 21, 25, and 70. Denote, with abuse of superscripting notation for simplicity of illustration, their learned feature representations by $\mathbf{z}^1, \mathbf{z}^{21}, \mathbf{z}^{25}, \mathbf{z}^{70}$. Smoothing-based regularization approaches would encourage, for example, \mathbf{z}^{21} to be similar to \mathbf{z}^{25} . However, we might also like \mathbf{z}^{21} to be somewhat similar to \mathbf{z}^1 , yet not as similar as \mathbf{z}^{21} is to \mathbf{z}^{25} . Let $\sigma(\cdot, \cdot)$ denote a similarity function over vectors, such as the cosine similarity. Restating our preference above, we would like $\sigma(\mathbf{z}^{21}, \mathbf{z}^{25}) > \sigma(\mathbf{z}^{21}, \mathbf{z}^1)$. Extending the idea, we can construct a complete desired ordering based on the distance in label space (age) from the anchor \mathbf{z}^{21} : $\sigma(\mathbf{z}^{21}, \mathbf{z}^{25}) > \sigma(\mathbf{z}^{21}, \mathbf{z}^1) > \sigma(\mathbf{z}^{21}, \mathbf{z}^{70})$. This construction can be repeated for the other samples as anchors. For example, using \mathbf{z}^{25} as anchor: $\sigma(\mathbf{z}^{25}, \mathbf{z}^{21}) > \sigma(\mathbf{z}^{25}, \mathbf{z}^1) > \sigma(\mathbf{z}^{25}, \mathbf{z}^{70})$. In general, these conditions encode an inductive bias that, for a given input sample, items that are closer in label space are also closer in feature space.

The rest of this paper is structured as follows. We next provide an overview of related work in deep imbalanced classification and regression. We then describe the RankSim regularizer for deep imbalanced regression, which encodes a more “global” inductive bias on label-space/feature-space relationships than state-of-the-art smoothing approaches: RankSim’s inductive bias captures not only nearby relationships but also distant relationships in label and feature space. Finally, we present new state-of-the-art results on three public benchmarks for deep imbalanced regression, as well as ablation experiments and analysis. Code and pretrained weights are available at <https://github.com/BorealisAI/ranksim-imbalanced-regression>.

2. Related Work

Imbalanced Classification. Prior work on learning with imbalanced data has predominantly focused on the imbalanced (or long-tailed) classification problem, where many classes have very few instances. Approaches can be grouped into data-based and model-based paradigms.

Data-based methods are associated with the input data. One strategy is re-sampling—over-sampling the minority classes (Chawla et al., 2002; Byrd & Lipton, 2019) or under-sampling the majority classes (Han et al., 2005; Buda et al., 2018). Recent work (Chang et al., 2021) introduced the re-sampling strategy to imbalanced object detection with a novel object-centric memory replay framework. Another option is data augmentation. Implicit semantic data augmentation (ISDA) (Wang et al., 2019) estimates the covariance matrices of each class to obtain semantic directions and generate augmented samples; Remix (Chou et al., 2020) and UniMix (Xu et al., 2021) extend mixup (Zhang et al., 2018) to enhance minority classes. *Model-based* methods address the imbalance problem from a model perspective. Logit adjustment (Tian et al., 2020; Tang et al., 2020) relies on post-hoc correction to adjust the decision boundary after training. Cost-sensitive learning (Lin et al., 2017; Cao et al., 2019; Ren et al., 2020) aims at adjusting losses for different classes during training, considering that the costs caused by different classes are not equal. For example, focal loss (Lin et al., 2017) inversely re-weights the loss with the prediction probabilities based on the observation that the minority classes suffer from achieving low loss values. Representation learning approaches (Yang & Xu, 2020; Kang et al., 2021; Jiang et al., 2021) primarily help the model focus on learning a less biased feature extractor. Two-stage (or decoupled) training, introduced by Kang et al. (2020), splits the learning procedure into separate stages for representation learning and classifier learning. A recent work (Zhong et al., 2021) introduced label-aware smoothing to handle different degrees of over-confidence for classes and improve classifier learning in the second stage.

Imbalanced Regression. The continuity in label space makes imbalanced regression different from imbalanced classification. On the one hand, the target values can be infinite and boundless, which makes many methods designed for deep imbalanced classification untenable. On the other hand, the continuity in label space can provide extra information about data instance relationships.

Early work (Chawla et al., 2002; Torgo et al., 2013; Branco et al., 2017) on imbalanced regression attempted to re-sample the training set by synthesizing new samples for minority targets. To build meaningful connections between different labels and handle potential missing values, state-of-the-art approaches proposed to use kernel density es-

timation to perform smoothing on the target distribution. Label distribution smoothing (LDS) (Yang et al., 2021) and DenseLoss (Steininger et al., 2021) share the similar idea of applying a Gaussian kernel to the empirical label density to estimate an “effective” label density distribution that takes the continuity of labels into account. Feature distribution smoothing (FDS) (Yang et al., 2021), aiming to introduce continuity of the feature, performs distribution smoothing on the feature space by transferring the feature statistics (mean and covariance) between nearby targets. However, these methods only account for the nearby label values, i.e. they encode a “local” inductive bias. In contrast, RankSim captures both nearby and distant relationships in label and feature space, encoding a more “global” inductive bias.

Some methods for imbalanced classification can also be adapted to regression. For example, inspired by Focal loss (Lin et al., 2017), a regression variant Focal-R (Yang et al., 2021) re-weights the loss values by the continuous L_1 error. We will show in the experiments that RankSim is complementary to Focal-R and other conventional imbalanced learning techniques. Progressive Margin Loss (PML) (Deng et al., 2021) focuses on imbalanced age estimation. It constructs the age relationship by learning label distributions, and handles the imbalanced data with margin-based imbalanced classification techniques.

Ranking-based Methods. The optimization of ranking-based losses is challenging due to the non-differentiability of the ranking operation. To approximate the non-differentiable and non-decomposable ranking operation, Taylor et al. (2008) considered the expectation of the ranking under a random perturbation; Song et al. (2016) leveraged the dynamic programming algorithm to update the weights; He et al. (2018), Cakir et al. (2019) and Revaud et al. (2019) optimized a differentiable relaxation of histogram binning (Ustinova & Lempitsky, 2016); Engilberge et al. (2019) proposed to learn an LSTM-based surrogate network; Cuturi et al. (2019) formulated it as an optimal transport problem. In our implementation, we recast the ranking operation as the minimizer of a linear combinatorial objective (Rolínek et al., 2020) and leverage an elegant method for efficient backpropagation through blackbox combinatorial solvers (Vlastelica et al., 2020) (details in Section 3). While prior work focused on directly optimizing ranking-based metrics such as Average Precision (AP) or Normalized Discounted Cumulative Gain (NDCG), we aim to make the ranking of feature-space neighbors to be similar to the ranking of label-space neighbors.

3. Method

To formalize the proposed regularizer, we first set up some preliminaries. Let $\mathbf{a} \in \mathbb{R}^n$ be an arbitrary vector of n

real values. Let rk denote the ranking function, such that $\text{rk}(\mathbf{a})$ is the permutation of $\{1, \dots, n\}$ containing the rank (in sorted order) of each element in \mathbf{a} . That is, the i th element in $\text{rk}(\mathbf{a})$ is given by

$$\text{rk}(\mathbf{a})_i = 1 + |\{j : \mathbf{a}_j > \mathbf{a}_i\}|. \quad (1)$$

For example, if $\mathbf{a} = [9, 5, 11, 6]$, then $\text{rk}(\mathbf{a}) = [2, 4, 1, 3]$. $\text{rk}(\mathbf{a})_1$ is 2 because \mathbf{a}_1 is the second-largest (rank = 2) element in \mathbf{a} ; $\text{rk}(\mathbf{a})_2$ is 4 because \mathbf{a}_2 is the fourth-largest element in \mathbf{a} ; $\text{rk}(\mathbf{a})_3$ is 1 because \mathbf{a}_3 is the largest element in \mathbf{a} ; finally, $\text{rk}(\mathbf{a})_4$ is 3 because \mathbf{a}_4 is the third-largest element in \mathbf{a} .

Let our regression dataset consist of a set of pairs (\mathbf{x}_i, y_i) , where \mathbf{x}_i denotes the input and y_i the corresponding continuous-value target¹. Denote by $\mathbf{z} = f(\mathbf{x}; \theta)$ the feature representation of \mathbf{x} generated by a neural network parameterized by θ . The idea behind RankSim (ranking similarity) regularization is to encourage alignment between the ranking of neighbors in label space (the y ’s) and the ranking of neighbors in feature space (the \mathbf{z} ’s).

Consider a subset of pairs $\mathcal{M} = \{(\mathbf{x}_i, y_i), i = 1, \dots, M\}$. Let $\mathbf{S}^y \in \mathbb{R}^{M \times M}$ be the pairwise similarity matrix obtained by applying similarity function σ^y in label space across all elements in \mathcal{M} . In other words, the (i, j) th entry in \mathbf{S}^y is given by

$$\mathbf{S}_{i,j}^y = \sigma^y(y_i, y_j), \quad (2)$$

which is the similarity between items i and j in label space. For continuous scalar labels, σ^y can simply return the negative absolute distance. Analogously, let $\mathbf{S}^z \in \mathbb{R}^{M \times M}$ be the pairwise similarity matrix obtained by applying similarity function σ^z in feature space. The (i, j) th entry in \mathbf{S}^z is given by

$$\mathbf{S}_{i,j}^z = \sigma^z(\mathbf{z}_i, \mathbf{z}_j) = \sigma^z(f(\mathbf{x}_i; \theta), f(\mathbf{x}_j; \theta)), \quad (3)$$

which is the similarity between items i and j in feature space. σ^z is a similarity function defined over vectors, such as the cosine similarity.

We can now define the RankSim regularization loss with respect to subset \mathcal{M} :

$$\mathcal{L}_{\text{RankSim}} = \sum_{i=1}^{|\mathcal{M}|} \ell(\text{rk}(\mathbf{S}_{[i,:]}^y), \text{rk}(\mathbf{S}_{[i,:]}^z)), \quad (4)$$

where $[i, :]$ denotes the i th row in the matrix and ranking similarity function ℓ penalizes differences in the input vectors. Concretely, we adopt the mean squared error for ℓ , which makes minimizing ℓ equivalent to maximizing the

¹In the following manuscript, the *target* is used interchangeably with the *label*.

Spearman correlation of the label-space and feature-space ranking vectors. During training, \mathcal{M} is constructed from each batch; to reduce ties and boost the relative representation of infrequent labels, we sample \mathcal{M} from the current batch such that each label occurs at most once.

Intuitively, Eq. 4 says that, given an input (\mathbf{x}_i, y_i) , we would like the sorted list of its neighbors in label space to match the sorted list of its neighbors in feature space as closely as possible. When the match is exact, the loss is zero. Using the previous age estimation example, since $\sigma^y(y^{11}, y^9) > \sigma^y(y^{11}, y^6) > \sigma^y(y^{11}, y^5) > \sigma^y(y^{11}, y^{55}) > \sigma^y(y^{11}, y^{60})$, where σ^y is the negative absolute distance, we would like $\sigma^z(\mathbf{z}^{11}, \mathbf{z}^9) > \sigma^z(\mathbf{z}^{11}, \mathbf{z}^6) > \sigma^z(\mathbf{z}^{11}, \mathbf{z}^5) > \sigma^z(\mathbf{z}^{11}, \mathbf{z}^{55}) > \sigma^z(\mathbf{z}^{11}, \mathbf{z}^{60})$, where σ^z is the cosine similarity, for example.

Eq. 4 is challenging to optimize because of the non-differentiability of the ranking operation. Ranking-based losses are piecewise constant functions of their input, and the gradient is zero almost everywhere: intuitively, small changes in the input to the ranking function do not always result in a change in the output ranking. However, we can recast the ranking operation as the minimizer of a linear combinatorial objective (Rolínek et al., 2020):

$$\mathbf{rk}(\mathbf{a}) = \arg \min_{\pi \in \Pi_n} \mathbf{a} \cdot \pi, \quad (5)$$

where Π_n is the set of all permutations of $\{1, \dots, n\}$. This allows us leverage an elegant method for efficient backpropagation through blackbox combinatorial solvers (Vlastelica et al., 2020). To obtain an informative gradient from the piecewise constant loss landscape, (Vlastelica et al., 2020) implicitly constructs a family of piecewise affine continuous interpolation functions parameterized by a single hyperparameter $\lambda > 0$ that trades off the informativeness of the gradient with fidelity to the original function. During the backward pass, instead of returning the true gradient (zero almost everywhere), we compute and return the gradient of the continuous interpolation:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{a}} = -\frac{1}{\lambda} (\mathbf{rk}(\mathbf{a}) - \mathbf{rk}(\mathbf{a}_\lambda)), \quad (6)$$

where \mathbf{a}_λ is constructed based on the incoming gradient information $\frac{\partial \mathcal{L}}{\partial \mathbf{rk}}$ by

$$\mathbf{a}_\lambda = \mathbf{a} + \lambda \frac{\partial \mathcal{L}}{\partial \mathbf{rk}}. \quad (7)$$

We can therefore backpropagate through the ranking operations in Eq. 4 at the cost of an additional call to \mathbf{rk} (i.e., the call in Eq. 6 on the perturbed input). For clarity, the blackbox “solver” in our case is simply the ranking function \mathbf{rk} , which can be implemented by sorting operations; there is no need for a general combinatorial solver.

The resulting regularizer is straightforward to implement, can be computed in closed form, and introduces only two hyperparameters: the interpolation strength λ , and the balancing weight γ on the regularization term in the overall network loss (i.e., adding $\gamma \mathcal{L}_{\text{RankSim}}$ to the other loss terms).

4. Experiments

We performed extensive experimental validation of RankSim on three public benchmarks for deep imbalanced regression. We first describe the benchmarks, metrics, and baselines. We then present experimental results on the three benchmarks, including comparisons with state-of-the-art approaches. Finally, we present ablation studies and analysis to better understand the impact of our design choices.

Benchmarks. We consider three datasets recently introduced by Yang et al. (2021). *IMDB-WIKI-DIR* is an age estimation dataset derived from *IMDB-WIKI* (Rothe et al., 2016), which consists of face images with age annotations. It has 191,509 training samples, 11,022 validation samples and 11,022 test samples; *AgeDB-DIR* is an age estimation dataset derived from *AgeDB* (Moschoglou et al., 2017), with 12,208 samples for training, 2,140 samples for validation, and 2,140 samples for testing; *STS-B-DIR* is a natural language dataset derived from the *STS-B* (Cer et al., 2017; Wang et al., 2018), which provides (continuous) similarity score between sentence pairs. It contains 5,249 training sentence pairs, 1,000 validation pairs and 1,000 test pairs. For further details on the benchmarks, including label distribution plots, we refer the interested reader to Yang et al. (2021). All datasets contain imbalanced training set and balanced validation/test sets.

Evaluation Metrics. Following Yang et al. (2021) and common practice in imbalanced learning (Liu et al., 2019), we report overall results on the whole test set, as well as on the subsets of *many-shot region* (bins with >100 training samples), *medium-shot region* (bins with 20 to 100 training samples), and *few-shot region* (bins with <20 training samples). In *IMDB-WIKI-DIR* and *AgeDB-DIR*, each bin is 1 year. In *STS-B-DIR*, the bin size is 0.1. We report mean absolute error (MAE, lower is better) and geometric mean (GM, lower is better) on *IMDB-WIKI-DIR* and *AgeDB-DIR*, and mean squared error (MSE, lower is better) and Pearson correlation (higher is better) on *STS-B-DIR*. Additional metrics are provided in the supplementary.

Baselines. For fair comparison, we adopt the standard network architectures specified in Yang et al. (2021), i.e., ResNet-50 for *IMDB-WIKI-DIR* and *AgeDB-DIR*, and BiLSTM + GloVe word embeddings for *STS-B-DIR*. RankSim is orthogonal to conventional imbalanced learning techniques, such as re-weighting (Lin et al., 2017) and two-stage (or

Table 1. Results on *IMDB-WIKI-DIR*. Baseline numbers are quoted from (Yang et al., 2021). The best results for each method (Vanilla, Focal-R, RRT, SQINV) are in **bold**. The best results for each metric and data subset (entire column) are in **bold and red**.

	MAE ↓				GM ↓			
	All	Man.	Med.	Few	All	Man.	Med.	Few
VANILLA	8.06	7.23	15.12	26.33	4.57	4.17	10.59	20.46
+ RANKSIM	7.72	6.93	14.48	25.38	4.27	3.90	10.02	17.84
+ LDS	7.83	7.31	12.43	22.51	4.42	4.19	7.00	13.94
+ LDS + RANKSIM	7.57	7.00	12.16	22.44	4.23	4.00	6.81	13.23
+ FDS	7.85	7.18	13.35	24.12	4.47	4.18	8.18	15.18
+ FDS + RANKSIM	7.74	6.93	14.71	24.91	4.34	3.96	10.35	16.85
+ LDS + FDS	7.78	7.20	12.61	22.19	4.37	4.12	7.39	12.61
+ LDS + FDS + RANKSIM	7.69	7.13	12.30	21.43	4.34	4.13	6.72	12.48
FOCAL-R	7.97	7.12	15.14	26.96	4.49	4.10	10.37	21.20
+ RANKSIM	7.77	6.99	14.23	26.01	4.38	4.03	9.25	20.16
+ LDS	7.90	7.10	14.72	25.84	4.47	4.09	10.11	19.14
+ LDS + RANKSIM	7.71	6.99	13.65	24.97	4.31	3.98	8.72	17.56
+ FDS	7.96	7.14	14.71	26.06	4.51	4.12	10.16	19.56
+ FDS + RANKSIM	7.75	7.01	14.06	24.56	4.33	3.99	9.04	16.26
+ LDS + FDS	7.88	7.10	14.08	25.75	4.47	4.11	9.32	18.67
+ LDS + FDS + RANKSIM	7.67	6.91	14.07	25.01	4.28	3.93	9.38	18.41
RRT	7.81	7.07	14.06	25.13	4.35	4.03	8.91	16.96
+ RANKSIM	7.55	6.83	13.47	24.72	4.17	3.86	8.66	15.54
+ LDS	7.79	7.08	13.76	24.64	4.34	4.02	8.72	16.92
+ LDS + RANKSIM	7.56	6.83	13.06	24.78	4.23	3.91	8.55	17.44
+ FDS	7.65	7.02	12.68	23.85	4.31	4.03	7.58	16.28
+ FDS + RANKSIM	7.35	6.81	11.50	22.75	4.05	3.85	6.05	14.68
+ LDS + FDS	7.65	7.06	12.41	23.51	4.31	4.07	7.17	15.44
+ LDS + FDS + RANKSIM	7.37	6.80	11.83	23.11	4.06	3.84	6.33	14.71
SQINV	7.87	7.24	12.44	22.76	4.47	4.22	7.25	15.10
+ RANKSIM	7.42	6.84	12.12	22.13	4.10	3.87	6.74	12.78
+ LDS	7.83	7.31	12.43	22.51	4.42	4.19	7.00	13.94
+ LDS + RANKSIM	7.57	7.00	12.16	22.44	4.23	4.00	6.81	13.23
+ FDS	7.83	7.23	12.60	22.37	4.42	4.20	6.93	13.48
+ FDS + RANKSIM	7.50	6.93	12.09	21.68	4.19	3.97	6.65	13.28
+ LDS + FDS	7.78	7.20	12.61	22.19	4.37	4.12	7.39	12.61
+ LDS + FDS + RANKSIM	7.69	7.13	12.30	21.43	4.34	4.13	6.72	12.48
OURS vs. VANILLA	+0.71	+0.43	+3.62	+4.90	+0.52	+0.33	+4.54	+7.98
OURS vs. Yang et al.	+0.30	+0.22	+0.91	+0.76	+0.26	+0.18	+0.88	+0.13

decoupled) training (Kang et al., 2020), as well as to state-of-the-art distribution smoothing approaches for deep imbalanced regression (Yang et al., 2021). We apply RankSim to regularize standard methods including Vanilla, Focal-R, RRT, SQINV (or INV), and apply the state-of-the-art LDS and FDS (Yang et al., 2021). The full details about methods and training are described in the supplementary A. We present detailed experimental analysis verifying that, not only does RankSim lift the state-of-the-art performance on the three benchmarks, it is in fact *complementary* to these well-established approaches.

4.1. IMDB-WIKI-DIR

Table 1 shows experimental results on the *IMDB-WIKI-DIR* benchmark. The baseline numbers are quoted from Yang et al. (2021). The table is grouped into four sections: vanilla (base network), Focal-R (focal loss (Lin et al., 2017) adapted to regression), regressor retraining (two-stage training (Kang et al., 2020) adapted to regression, abbreviated RRT), and square-root inverse frequency re-weighting (SQINV). Within each group, we first show the results obtained by applying the state-of-the-art label and feature distribution smoothing methods (LDS and FDS, respectively)

Table 2. Results on *AgeDB-DIR*. Baseline numbers are quoted from (Yang et al., 2021). The best results for each method (Vanilla, Focal-R, RRT and SQINV) are in **bold**. The best results for each metric and data subset (entire column) are in **bold and red**.

	MAE ↓				GM ↓			
	All	Man.	Med.	Few	All	Man.	Med.	Few
VANILLA	7.77	6.62	9.55	13.67	5.05	4.23	7.01	10.75
+ RANKSIM	7.13	6.51	8.17	10.12	4.48	4.01	5.27	6.79
+ LDS	7.67	6.98	8.86	10.89	4.85	4.39	5.80	7.45
+ LDS + RANKSIM	6.99	6.38	7.88	10.23	4.40	3.97	5.30	6.93
+ FDS	7.55	6.50	8.97	13.01	4.75	4.03	6.42	9.93
+ FDS + RANKSIM	7.33	6.49	8.53	11.98	4.82	4.19	6.16	8.99
+ LDS + FDS	7.55	7.01	8.24	10.79	4.72	4.36	5.45	6.79
+ LDS + FDS + RANKSIM	7.03	6.54	7.68	9.92	4.45	4.07	5.23	6.35
FOCAL-R	7.64	6.68	9.22	13.00	4.90	4.26	6.39	9.52
+ RANKSIM	7.15	6.45	7.97	11.50	4.53	4.10	5.10	8.50
+ LDS	7.56	6.67	8.82	12.40	4.82	4.27	5.87	8.83
+ LDS + RANKSIM	7.25	6.40	8.71	11.24	4.58	4.02	5.99	7.52
+ FDS	7.65	6.89	8.70	11.92	4.83	4.32	5.89	8.04
+ FDS + RANKSIM	7.25	6.72	7.86	10.58	4.54	4.22	4.84	7.57
+ LDS + FDS	7.47	6.69	8.30	12.55	4.71	4.25	5.36	8.59
+ LDS + FDS + RANKSIM	7.09	6.17	8.71	11.68	4.46	3.85	5.76	8.78
RRT	7.74	6.98	8.79	11.99	5.00	4.50	5.88	8.63
+ RANKSIM	7.11	6.53	8.00	10.04	4.52	4.19	5.05	6.77
+ LDS	7.72	7.00	8.75	11.62	4.98	4.54	5.71	8.27
+ LDS + RANKSIM	6.94	6.43	7.54	10.10	4.37	3.97	5.11	7.05
+ FDS	7.70	6.95	8.76	11.86	4.82	4.32	5.83	8.08
+ FDS + RANKSIM	7.11	6.55	7.99	10.02	4.49	4.13	5.13	6.85
+ LDS + FDS	7.66	6.99	8.60	11.32	4.80	4.42	5.53	6.99
+ LDS + FDS + RANKSIM	7.13	6.54	8.07	10.12	4.55	4.18	5.20	6.87
SQINV	7.81	7.16	8.80	11.20	4.99	4.57	5.73	7.77
+ RANKSIM	6.91	6.34	7.79	9.89	4.28	3.92	4.88	6.89
+ LDS	7.67	6.98	8.86	10.89	4.85	4.39	5.80	7.45
+ LDS + RANKSIM	6.99	6.38	7.88	10.23	4.40	3.97	5.30	6.90
+ FDS	7.69	7.10	8.86	9.98	4.83	4.41	5.97	6.29
+ FDS + RANKSIM	7.02	6.49	7.84	9.68	4.53	4.13	5.37	6.89
+ LDS + FDS	7.55	7.01	8.24	10.79	4.72	4.36	5.45	6.79
+ LDS + FDS + RANKSIM	7.03	6.54	7.68	9.92	4.45	4.07	5.23	6.35
OURS vs. VANILLA	+0.86	+0.45	+2.01	+3.99	+0.77	+0.38	+2.17	+4.40
OURS vs. Yang et al.	+0.56	+0.33	+0.70	+0.30	+0.43	+0.18	+0.52	-0.06

(Yang et al., 2021), both separately and in combination. We integrated RankSim on top of each of these baselines.

Within each of the four sections, the best results are highlighted in bold. RankSim consistently obtains the best result within each section, across both metrics (MAE, GM) and test subsets (all, many-shot, medium-shot, few-shot). This outcome indicates that RankSim is complementary to standard imbalanced learning techniques.

For each metric and test subset, the best overall result is highlighted in red. RankSim sets a new state-of-the-art across all metrics and test subsets. For example, when combined with two-stage retraining (RRT) and feature distribution smoothing (FDS), RankSim achieves a leading mean absolute error of 7.35 on the overall test set. In the few-shot region, RankSim combined with re-weighting (SQINV), label distribution smoothing (LDS), and feature distribution smoothing (FDS) achieves a leading mean absolute error of 21.43.

4.2. AgeDB-DIR

Table 2 shows experimental results on the *AgeDB-DIR* benchmark, following the same structure as *IMDB-WIKI-*

Table 3. Results on STS-B-DIR. Baseline numbers are quoted from (Yang et al., 2021). The best results for each method (Vanilla, Focal-R, RRT and INV) are in bold. The best results for each metric and data subset (entire column) are in bold and red.

	MSE ↓				Pearson cor. (%) ↑			
	All	Man.	Med.	Few	All	Man.	Med.	Few
VANILLA	0.974	0.851	1.520	0.984	74.2	72.0	62.7	75.2
+ RANKSIM	0.873	0.908	0.767	0.705	76.8	71.0	72.9	85.2
+ LDS	0.914	0.819	1.319	0.955	75.6	73.4	63.8	76.2
+ LDS + RANKSIM	0.889	0.911	0.849	0.690	76.2	70.7	70.0	85.6
+ FDS	0.916	0.875	1.027	1.086	75.5	73.0	67.0	72.8
+ FDS + RANKSIM	0.884	0.924	0.767	0.685	76.5	70.4	72.5	85.7
+ LDS + FDS	0.907	0.802	1.363	0.942	76.0	74.0	65.2	76.6
+ LDS + FDS + RANKSIM	0.903	0.908	0.911	0.804	75.8	70.6	69.0	82.7
FOCAL-R	0.951	0.843	1.425	0.957	74.6	72.3	61.8	76.4
+ RANKSIM	0.887	0.889	0.918	0.745	76.2	70.8	70.4	84.6
+ LDS	0.930	0.807	1.449	0.993	75.7	73.9	62.4	75.4
+ LDS + RANKSIM	0.872	0.887	0.847	0.718	76.7	71.2	70.3	85.1
+ FDS	0.920	0.855	1.169	1.008	75.1	72.6	66.4	74.7
+ FDS + RANKSIM	0.913	0.952	0.793	0.723	75.6	69.6	71.5	84.8
+ LDS + FDS	0.940	0.849	1.358	0.916	74.9	72.2	66.3	77.3
+ LDS + FDS + RANKSIM	0.911	0.943	0.779	0.866	75.7	69.9	71.4	81.2
RRT	0.964	0.842	1.503	0.978	74.5	72.4	62.3	75.4
+ RANKSIM	0.865	0.876	0.867	0.670	77.1	72.2	68.3	86.1
+ LDS	0.916	0.817	1.344	0.945	75.7	73.5	64.1	76.6
+ LDS + RANKSIM	0.874	0.893	0.833	0.722	77.0	72.3	68.3	84.8
+ FDS	0.929	0.857	1.209	1.025	74.9	72.1	67.2	74.0
+ FDS + RANKSIM	0.871	0.874	0.898	0.734	76.8	72.0	68.7	84.5
+ LDS + FDS	0.903	0.806	1.323	0.936	76.0	73.8	65.2	76.7
+ LDS + FDS + RANKSIM	0.882	0.892	0.887	0.702	76.6	71.7	68.0	85.5
INV	1.005	0.894	1.482	1.046	72.8	70.3	62.5	73.2
+ RANKSIM	1.091	1.056	1.240	1.118	69.9	65.2	60.1	76.0
+ LDS	0.914	0.819	1.319	0.955	75.6	73.4	63.8	76.2
+ LDS + RANKSIM	0.889	0.911	0.849	0.690	76.2	70.7	70.0	85.6
+ FDS	0.927	0.851	1.225	1.012	75.0	72.4	66.6	74.2
+ FDS + RANKSIM	1.083	1.035	1.301	1.063	70.0	64.8	68.9	76.7
+ LDS + FDS	0.907	0.802	1.363	0.942	76.0	74.0	65.2	76.6
+ LDS + FDS + RANKSIM	0.903	0.908	0.911	0.804	75.8	70.6	69.0	82.7
OURS vs. VANILLA	+0.109	-0.023	+0.753	+0.314	+2.9	+0.3	+10.2	+10.9
OURS vs. Yang et al.	+0.038	-0.072	+0.260	+0.246	+1.1	-1.7	+5.7	+8.8

DIR above. Comparing within each section (vanilla, Focal-R, RRT, SQINV), RankSim obtains the best result in 31 out of the 32 combinations of metric (MAE, GM) and test subset (all, many-shot, medium-shot, few-shot), again demonstrating that RankSim is complementary to standard imbalanced learning techniques. Overall, RankSim sets a new state-of-the-art on 7 out of 8 metric-subset combinations². For example, when combined with square-root inverse frequency re-weighting, RankSim achieves a leading mean absolute error of 6.91 on the overall test set. In the few-shot setting, RankSim with square-root inverse frequency re-weighting and feature distribution smoothing achieves a leading mean absolute error of 9.68.

4.3. STS-B-DIR

Table 3 shows experimental results on the STS-B-DIR benchmark. INV refers to inverse frequency re-weighting. RankSim consistently achieves state-of-the-art performance on the test set overall, as well as in the medium-shot and few-shot categories; the INV+LDS+FDS baseline performs

²In our experiments, we observed that a smaller batch size than recommended in Yang et al. (2021) improved baseline performance. We include these improved baseline numbers in the supplementary. RankSim outperforms the baselines in both settings.

Table 4. Different choices of ranking similarity penalty function ℓ in Eq. 4. Results are on AgeDB-DIR (top) and IMDB-WIKI-DIR (bottom) with SQINV. MSE is used in all other experiments.

	MAE ↓				GM ↓			
	All	Many	Med.	Few	All	Many	Med.	Few
<i>AgeDB-DIR</i>								
COSINE DISTANCE	6.99	6.55	7.61	9.47	4.40	4.14	4.84	5.95
HUBER	7.05	6.43	7.99	10.25	4.53	4.07	5.68	6.69
L_∞	7.07	6.53	7.81	10.08	4.60	4.27	5.07	7.06
MAE	7.04	6.42	7.94	10.42	4.37	3.95	5.18	7.27
MSE	6.91	6.34	7.79	9.89	4.28	3.92	4.88	6.89
<i>IMDB-WIKI-DIR</i>								
COSINE DISTANCE	7.38	6.79	12.25	21.87	4.07	3.83	7.14	13.62
HUBER	7.46	6.87	12.22	22.87	4.17	3.94	6.17	14.07
L_∞	7.57	7.01	11.99	22.28	4.29	4.08	6.54	13.59
MAE	7.48	6.86	12.50	23.13	4.14	3.89	7.07	14.79
MSE	7.42	6.84	12.12	22.13	4.10	3.87	6.74	12.78

Table 5. Different choices of feature similarity function σ^z . Results are on AgeDB-DIR (top) and IMDB-WIKI-DIR (bottom) with SQINV. Cosine similarity is used in all other experiments.

	MAE ↓				GM ↓			
	All	Many	Med.	Few	All	Many	Med.	Few
<i>AgeDB-DIR</i>								
NEGATIVE L_∞	7.14	6.56	7.92	10.37	4.51	4.15	5.11	7.05
NEGATIVE MAE	7.25	6.54	8.40	10.77	4.67	4.18	5.64	7.89
NEGATIVE MSE	7.06	6.61	7.80	9.20	4.43	4.17	4.95	5.83
CORRELATION SIMILARITY	7.32	6.77	8.24	10.01	4.75	4.43	5.37	6.44
COSINE SIMILARITY	6.91	6.34	7.79	9.89	4.28	3.92	4.88	6.89
<i>IMDB-WIKI-DIR</i>								
NEGATIVE L_∞	7.44	6.85	12.10	23.33	4.15	3.92	6.61	15.14
NEGATIVE MAE	7.48	6.92	11.86	22.56	4.21	3.99	6.56	14.33
NEGATIVE MSE	7.46	6.87	12.26	22.20	4.17	3.93	6.97	14.55
CORRELATION SIMILARITY	7.78	7.03	14.09	24.97	4.26	3.93	9.00	16.59
COSINE SIMILARITY	7.42	6.84	12.12	22.13	4.10	3.87	6.74	12.78

slightly better in the many-shot category. The best overall results are obtained by RankSim combined with two-stage retraining (RRT): 0.865 mean squared error and 77.1% Pearson correlation. In the few-shot setting, RankSim combined with RRT achieves a leading 0.670 mean squared error and 86.1% Pearson correlation.

4.4. Ablations and Analysis

Different choices for ℓ in Eq. 4. RankSim uses the function ℓ in Eq. 4 to penalize differences between the ranking of neighbors in label space and the ranking of neighbors in feature space. We adopted mean squared error in all of the benchmark experiments. In this ablation study, to provide a more complete picture of different ranking similarity losses, we evaluate several other options for ℓ . Let \mathbf{rk}^a and \mathbf{rk}^b denote two m -dimensional ranking vectors of interest. We consider the following options for ℓ :

1) COSINE DISTANCE: $1 - \frac{\mathbf{rk}^a \cdot \mathbf{rk}^b}{\|\mathbf{rk}^a\|_2 \times \|\mathbf{rk}^b\|_2}$

- 2) HUBER LOSS: $\frac{1}{m} \sum_{i=1}^m \ell_i$, with δ default as 1.0,

$$\ell_i = \begin{cases} 0.5 \times (\mathbf{r}\mathbf{k}_i^{\mathbf{a}} - \mathbf{r}\mathbf{k}_i^{\mathbf{b}})^2, & \text{if } |\mathbf{r}\mathbf{k}_i^{\mathbf{a}} - \mathbf{r}\mathbf{k}_i^{\mathbf{b}}| < \delta \\ \delta \times (|\mathbf{r}\mathbf{k}_i^{\mathbf{a}} - \mathbf{r}\mathbf{k}_i^{\mathbf{b}}| - 0.5 \times \delta), & \text{otherwise} \end{cases}$$
- 3) L_∞ DISTANCE: $\max_i |\mathbf{r}\mathbf{k}_i^{\mathbf{a}} - \mathbf{r}\mathbf{k}_i^{\mathbf{b}}|$
- 4) MAE: $\frac{1}{m} \sum_{i=1}^m |\mathbf{r}\mathbf{k}_i^{\mathbf{a}} - \mathbf{r}\mathbf{k}_i^{\mathbf{b}}|$
- 5) MSE: $\frac{1}{m} \sum_{i=1}^m (\mathbf{r}\mathbf{k}_i^{\mathbf{a}} - \mathbf{r}\mathbf{k}_i^{\mathbf{b}})^2$

Table 4 presents ablation results on *AgeDB-DIR* and *IMDB-WIKI-DIR* with SQINV. On *AgeDB-DIR*, implementing RankSim with *any* of the above options for ℓ improves the baseline SQINV. MSE achieves the best performance on the dataset overall, as well as on the many-shot subset. Cosine distance achieves the best performance on the medium-shot and few-shot subsets. On *IMDB-WIKI-DIR*, cosine distance and MSE achieve comparable best performance overall and in the few-shot subset. Either option consistently improves the baseline SQINV in all settings.

Different choices for feature similarity function σ^z . The similarity function σ^z in Eq. 3 quantifies the similarity of two data samples in feature space. We adopted cosine similarity in all of the benchmark experiments. Here, we evaluate RankSim with different choices for the feature-space similarity function. Let \mathbf{z}_1 and \mathbf{z}_2 denote two d -dimensional feature vectors. We consider the following options for σ^z :

- 1) NEGATIVE L_∞ : $-\max_i |\mathbf{z}_{1,i} - \mathbf{z}_{2,i}|$
- 2) NEGATIVE MAE: $-\frac{1}{d} \sum_{i=1}^d |\mathbf{z}_{1,i} - \mathbf{z}_{2,i}|$
- 3) NEGATIVE MSE: $-\frac{1}{d} \sum_{i=1}^d (\mathbf{z}_{1,i} - \mathbf{z}_{2,i})^2$
- 4) CORRELATION SIMILARITY: $\frac{(\mathbf{z}_1 - \bar{\mathbf{z}}_1) \cdot (\mathbf{z}_2 - \bar{\mathbf{z}}_2)}{\|\mathbf{z}_1 - \bar{\mathbf{z}}_1\|_2 \times \|\mathbf{z}_2 - \bar{\mathbf{z}}_2\|_2}$
- 5) COSINE SIMILARITY: $\frac{\mathbf{z}_1 \cdot \mathbf{z}_2}{\|\mathbf{z}_1\|_2 \times \|\mathbf{z}_2\|_2}$

Table 5 presents ablation results on *AgeDB-DIR* and *IMDB-WIKI-DIR* with SQINV. On *AgeDB-DIR*, implementing RankSim with *any* of the above options for σ^z improves the baseline SQINV. Cosine similarity achieves the best performance on the dataset overall, as well as on the many-shot subset. Negative MSE achieves the best performance on the few-shot subset. On *IMDB-WIKI-DIR*, cosine similarity achieves the best performance overall, in the many-shot subset, and in the few-shot subset. Negative MAE, negative MSE, and cosine similarity consistently improve the baseline SQINV.

Ablation on batch sampling. As described in Section 3, during training, we sample \mathcal{M} from the current batch such that each label occurs at most once. The intent is to reduce ties and boost the relative representation of infrequent labels. In this ablation, we consider removing the sampling and using the entire batch as \mathcal{M} . Table 6 shows the ablation

Table 6. Ablation study on batch sampling. “With sampling” refers to sampling \mathcal{M} from the current batch such that each label occurs at most once. Results are on *IMDB-WIKI-DIR* and *AgeDB-DIR* with SQINV re-weighting, and *STS-B-DIR* with Vanilla. We apply sampling in all other experiments.

<i>AgeDB-DIR</i>	MAE ↓				GM ↓			
	All	Many	Med.	Few	All	Many	Med.	Few
WITH SAMPLING	6.91	6.34	7.79	9.89	4.28	3.92	4.88	6.89
WITHOUT SAMPLING	7.10	6.52	7.99	10.17	4.53	4.17	5.16	6.85

<i>IMDB-WIKI-DIR</i>	MAE ↓				GM ↓			
	All	Many	Med.	Few	All	Many	Med.	Few
WITH SAMPLING	7.42	6.84	12.12	22.13	4.10	3.87	6.74	12.78
WITHOUT SAMPLING	7.45	6.85	12.27	23.05	4.14	3.89	7.04	15.67

<i>STS-B-DIR</i>	MSE ↓				Pearson cor. (%) ↑			
	All	Many	Med.	Few	All	Many	Med.	Few
WITH SAMPLING	0.873	0.908	0.767	0.705	76.8	71.0	72.9	85.2
WITHOUT SAMPLING	0.869	0.869	0.884	0.805	76.7	71.6	69.1	83.3

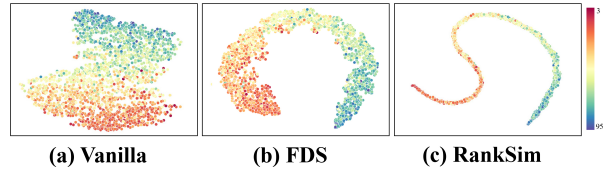


Figure 2. Qualitative visualization of ResNet-50 features on *AgeDB-DIR* balanced test set: (a) Vanilla, (b) Vanilla + FDS, and (c) Vanilla + RankSim. We use t-SNE by treating the continuous labels (age, from 3 to 95) as categorical (92 “classes”, with no samples of age 4). The label space is denoted with continuous spectral colors.

results. Applying sampling leads to better performance in the few-shot and medium-shot regions, and comparable performance overall.

Qualitative visualization of feature space. We extract the features from ResNet-50 (Vanilla, Vanilla with FDS, and Vanilla with RankSim) on the *AgeDB-DIR* test set and visualize using t-SNE technique in Fig. 2. To visually differentiate labels and show the continuity in feature space, the labels are denoted in continuous spectral colors (from red to blue). The visualization shows the continuity in features learned with FDS and RankSim.

Qualitative visualization of ranking matrices with balanced and imbalanced data. In Fig. 3, we visualize the average (batch-wise) label-space and feature-space ranking matrices under three training settings: vanilla network on balanced data, vanilla network on imbalanced data, and vanilla with RankSim on imbalanced data. We use *AgeDB-DIR* for this visualization; for the setting with balanced data, we extract a balanced subset of *AgeDB-DIR* consisting of the many-shot ages only (ages in the range of 23 to 63). For clarity of visualization, samples in each batch are pre-sorted

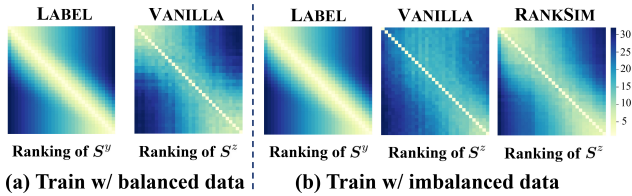


Figure 3. Average feature-space ranking matrices S^z , trained on (a) balanced subset of *AgeDB-DIR* (left two), and (b) imbalanced full *AgeDB-DIR* (right three). The corresponding label-space ranking matrix S^y is provided. For clarity of visualization, each batch is pre-sorted by label.

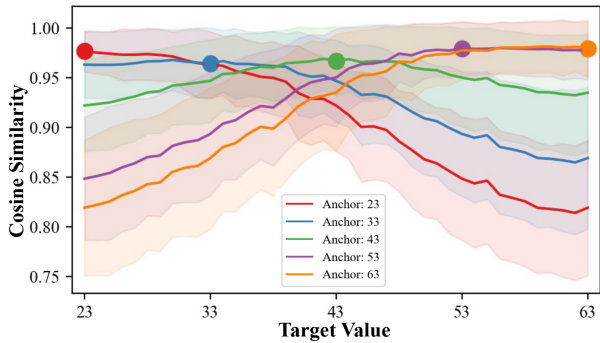


Figure 4. Feature cosine similarity, learned by Vanilla ResNet-50 from a balanced subset of *AgeDB-DIR*. Each curve depicts the feature similarity between the anchor and other labels (x-axis). E.g. the point (25, 0.974) on the red curve (anchor 23) indicates the mean feature cosine similarity between age 23 and 25 is 0.974. Each solid point is the mean cosine (intra) similarity between the data points of the corresponding anchor value.

by label (i.e., age). Each batch consists of 32 samples. The visualized matrices are obtained by applying the ranking operation rk on each row of S^y and S^z as in Eq. 4, and averaging the result over all test batches. For an example calculation, please refer to the supplementary.

Fig. 3(a) shows that the label-space and feature-space rankings tend to be consistent when training with balanced data. In other words, the sorted list of neighbors in label space tend to resemble the sorted list of neighbors in feature space. This validates RankSim’s inductive bias. Fig. 3(b) shows that this pattern disappears when training with imbalanced data. Training with the RankSim regularizer helps to recover the pattern: we can observe that the sorted list of neighbors in feature space (Fig. 3(b), right) again tends to resemble the sorted list of neighbors in label space (Fig. 3(b), left).

In Fig. 4, we additionally plot the cosine similarity between the vanilla network features on the balanced subset. The visualization provides an alternative view showing the consistency between label-space neighbors and feature-space neighbors when training on balanced data. For example, the curve for the youngest anchor shows a roughly mono-

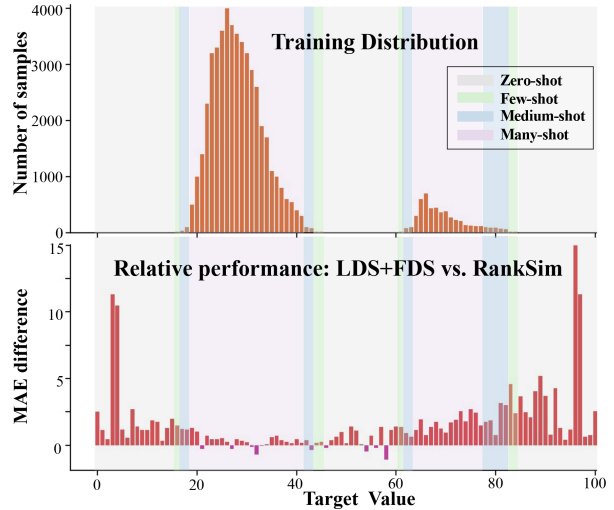


Figure 5. Training with zero-shot targets. The top figure is a hand-crafted subset of *IMDB-WIKI-DIR* that has zero-shot targets, imitating the zero-shot experiment construction in (Yang et al., 2021). The bottom figure is the MAE difference (LDS+FDS minus RankSim) for each target value on the balanced test set. We apply SQINV re-weighting to both LDS+FDS and RankSim. In the bottom figure, y-axis > 0 means that RankSim performs better.

tonic drop in the average feature similarity as we move to older ages. The curve for the middle (age 43) anchor shows roughly monotonic drops in the average feature similarity in both directions as we move to younger or older ages. The points on each curve are the mean similarity between the data instances of the anchor itself (e.g. the red point denotes the mean cosine similarity between all data of age 23). Thus, the solid points usually have highest similarity. Fig. 4 complements Fig. 3(a), showing from a feature centric view that, in the balanced setting, a vanilla network naturally learns representations such that items closer in label space are also closer in feature space.

Zero-shot targets. To demonstrate the ability of RankSim to handle the zero-shot scenario, we construct a subset of *IMDB-WIKI-DIR* that contains zero-shot targets. Fig. 5 (top) shows the hand-crafted distribution, which is intended to imitate the zero-shot experiment construction in Yang et al. (2021). Fig. 5 (bottom) shows that, in addition to performing comparably with LDS+FDS in or near the many-shot region, RankSim significantly outperforms LDS+FDS in the zero-shot region. The performance gap is largest for the zero-shot targets at the extremes. We surmise that the global ranking-driven representation learned by RankSim is effective at both interpolation and extrapolation, while LDS and FDS, as local smoothing-based methods, are primarily designed for interpolation.

Training cost. We measured the training time for *AgeDB-DIR* on four NVIDIA GeForce GTX 1080 Ti GPUs.

RankSim incurs a small overhead with respect to vanilla training and is faster to train than FDS: for one epoch, vanilla takes 12.2 seconds for a forward pass and 31.5 seconds for training; RankSim takes 16.8 seconds for a forward pass and 38.8 seconds for training; FDS takes 38.4 seconds for a forward pass and 60.5 seconds for training.

Sensitivity to hyperparameters. RankSim introduces two hyperparameters: the balancing weight γ and the interpolation strength λ . We performed sensitivity experiments on *AgeDB-DIR* and *IMDB-WIKI-DIR* in which we varied these two hyperparameters. The full results can be found in the supplementary. Performance is relatively robust to both γ and λ . Since RankSim requires a batch-wise calculation to construct the pairwise similarity matrices, we include in the supplementary additional sensitivity experiments in which we varied the batch size.

5. Conclusion

We introduced a novel regularizer for deep imbalanced regression that captures both nearby and distant relationships in label and feature space. RankSim achieves new state-of-the-art results on three benchmarks for deep imbalanced regression, and is complementary to conventional imbalanced learning techniques, including re-weighting, two-stage (decoupled) training, and label and feature distribution smoothing. We hope that our findings will unlock further advances in the training of regression networks with imbalanced data.

Limitations and future work. We have focused on training scenarios where the data samples are independent. In some applications, such as dense pixelwise regression in computer vision, inputs are coupled (e.g. pixels in an image are not independent) and the translation of the concept of neighbors in label space and feature space may require additional work. We leave this for a future extension. In addition, we plan to further study the generalization of the RankSim regularizer to representation learning more broadly.

References

- Branco, P., Targo, L., and Ribeiro, R. P. SMOGN: A pre-processing approach for imbalanced regression. In *International Workshop on Learning with Imbalanced Datasets: Theory and Applications*, 2017.
- Buda, M., Maki, A., and Mazurowski, M. A. A systematic study of the class imbalance problem in convolutional neural networks. *Neural networks*, 2018.
- Byrd, J. and Lipton, Z. C. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning*, 2019.
- Cakir, F., He, K., Xia, X., Kulis, B., and Sclaroff, S. Deep metric learning to rank. In *Conference on Computer Vision and Pattern Recognition*, 2019.
- Cao, K., Wei, C., Gaidon, A., Arechiga, N., and Ma, T. Learning imbalanced datasets with label-distribution-aware margin loss. In *Conference on Neural Information Processing Systems*, 2019.
- Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., and Specia, L. SemEval-2017 task 1: semantic textual similarity multilingual and crosslingual focused evaluation. In *International Workshop on Semantic Evaluation*, 2017.
- Chang, N., Yu, Z., Wang, Y.-X., Anandkumar, A., Fidler, S., and Alvarez, J. M. Image-level or object-level? a tale of two resampling strategies for long-tailed detection. In *International Conference on Machine Learning*, 2021.
- Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 2002.
- Chou, H.-P., Chang, S.-C., Pan, J.-Y., Wei, W., and Juan, D.-C. Remix: Rebalanced mixup. In *European Conference on Computer Vision Workshops*, 2020.
- Cuturi, M., Teboul, O., and Vert, J.-P. Differentiable ranking and sorting using optimal transport. In *Conference in Neural Information Processing Systems*, 2019.
- Deng, Z., Liu, H., Wang, Y., Wang, C., Yu, Z., and Sun, X. PML: Progressive margin loss for long-tailed age classification. In *Conference on Computer Vision and Pattern Recognition*, 2021.
- Engilberge, M., Chevallier, L., Pérez, P., and Cord, M. Sodeep: A sorting deep net to learn ranking loss surrogates. In *Conference on Computer Vision and Pattern Recognition*, 2019.
- Han, H., Wang, W., and Mao, B. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *International Conference in Intelligent Computing*, 2005.
- He, K., Cakir, F., Bargal, S. A., and Sclaroff, S. Hashing as tie-aware learning to rank. In *Conference on Computer Vision and Pattern Recognition*, 2018.
- Jiang, Z., Chen, T., Mortazavi, B., and Wang, Z. Self-damaging contrastive learning. In *International Conference on Machine Learning*, 2021.
- Kang, B., Xie, S., Rohrbach, M., Yan, Z., Gordo, A., Feng, J., and Kalantidis, Y. Decoupling representation and classifier for long-tailed recognition. In *International Conference on Learning Representations*, 2020.

- Kang, B., Li, Y., Xie, S., Yuan, Z., and Feng, J. Exploring balanced feature spaces for representation learning. In *International Conference on Learning Representations*, 2021.
- Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollar, P. Focal loss for dense object detection. In *International Conference on Computer Vision*, 2017.
- Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., and Yu, S. X. Large-scale long-tailed recognition in an open world. In *Conference on Computer Vision and Pattern Recognition*, 2019.
- Moschoglou, S., Papaioannou, A., Sagonas, C., Deng, J., Kotsia, I., and Zafeiriou, S. AgeDB: The first manually collected, in-the-wild age database. In *Conference on Computer Vision and Pattern Recognition Workshops*, 2017.
- Ren, J., Yu, C., Sheng, S., Ma, X., Zhao, H., Yi, S., and Li, H. Balanced meta-softmax for long-tailed visual recognition. In *Conference on Neural Information Processing Systems*, 2020.
- Revaud, J., Almazan, J., de Rezende, R. S., and de Souza, C. R. Learning with average precision: Training image retrieval with a listwise loss. In *International Conference on Computer Vision*, 2019.
- Rolínek, M., Musil, V., Paulus, A., Vlastelica, M., Michaelis, C., and Martius, G. Optimizing rank-based metrics with blackbox differentiation. In *Conference on Computer Vision and Pattern Recognition*, 2020.
- Rothe, R., Timofte, R., and Gool, L. V. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision*, 2016.
- Song, Y., Schwing, A. G., Zemel, R. S., and Urtasun, R. Training deep neural networks via direct loss minimization. In *International Conference on Machine Learning*, 2016.
- Steininger, M., Kobs, K., Davidson, P., Krause, A., and Hotho, A. Density-based weighting for imbalanced regression. *Machine Learning*, 2021.
- Tang, K., Huang, J., and Zhang, H. Long-tailed classification by keeping the good and removing the bad momentum causal effect. In *Conference on Neural Information Processing Systems*, 2020.
- Taylor, M., Guiver, J., Robertson, S., and Minka, T. SoftRank: Optimizing non-smooth rank metrics. In *International Conference on Web Search and Data Mining*, 2008.
- Tian, J., Liu, Y.-C., Glaser, N., Hsu, Y.-C., and Kira, Z. Posterior re-calibration for imbalanced datasets. In *Conference on Neural Information Processing Systems*, 2020.
- Torgo, L., Ribeiro, R. P., Pfahringer, B., and Branco, P. SMOTE for regression. In *Portuguese Conference on Artificial Intelligence*, 2013.
- Ustinova, E. and Lempitsky, V. Learning deep embeddings with histogram loss. In *Conference in Neural Information Processing Systems*, 2016.
- Vlastelica, M., Paulus, A., Musil, V., Martius, G., and Rolínek, M. Differentiation of blackbox combinatorial solvers. In *International Conference on Learning Representations*, 2020.
- Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Conference on Empirical Methods in Natural Language Processing*, 2018.
- Wang, Y., Pan, X., Song, S., Zhang, H., Huang, G., and Wu, C. Implicit semantic data augmentation for deep networks. In *Conference on Neural Information Processing Systems*, 2019.
- Xu, Z., Chai, Z., and Yuan, C. Towards calibrated model for long-tailed visual recognition from prior perspective. In *Conference on Neural Information Processing Systems*, 2021.
- Yang, Y. and Xu, Z. Rethinking the value of labels for improving class-imbalanced learning. In *Conference on Neural Information Processing Systems*, 2020.
- Yang, Y., Zha, K., Chen, Y.-C., Wang, H., and Katabi, D. Delving into deep imbalanced regression. In *International Conference on Machine Learning*, 2021.
- Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.
- Zhong, Z., Cui, J., Liu, S., and Jia, J. Improving calibration for long-tailed recognition. In *Conference on Computer Vision and Pattern Recognition*, 2021.

Supplementary Material

A. Baseline Details

In this section, we provide a detailed description of all models including RankSim and all other baselines. In Table 1, Table 2 and Table 3, we apply RankSim to regularize a variety of standard imbalanced learning methods. We also apply RankSim to the state-of-the-art LDS and FDS methods (Yang et al., 2021), and follow them to combine their method with other standard methods. As RankSim is orthogonal to conventional imbalanced learning techniques, we apply it to all baselines to provide a full picture of its effectiveness.

VANILLA. It refers to conventionally train the backbone network without any imbalanced learning methods, i.e. ResNet-50 for *IMDB-WIKI-DIR* and *AgeDB-DIR*, and BiLSTM+GloVe for *STS-B-DIR*.

FOCAL-R. Inspired by Focal loss (Lin et al., 2017) which adds a factor to standard cross entropy to focus on hard samples, Yang et al. (2021) proposed its regression version Focal-R: $\frac{1}{n} \sum_{i=1}^n \text{Sigmoid}(|\beta e_i|)^\gamma$, where e_i is the error for i -th sample, β and γ are hyperparameters. The scaling factor is now a continuous function that maps the absolute error into the range of $[0,1]$. It shares the same idea as Focal loss to focus on hard samples. L_1 distance is used for e_i in all experiments with Focal-R in this paper. β and γ are set to 0.2 and 1.0 for all experiments.

RRT. Yang et al. (2021) extended the standard re-training method to regression by decoupling the training of backbone feature extractor and regressor (originally classifier (Kang et al., 2020)) into two stages, and termed it as Regressor Re-training (RRT). In the first stage, the network is trained normally, and in the second stage only the regressor (the last FC layer) is re-trained with inverse re-weighting. As RankSim is a regularizer for the feature learning, we apply it in the first stage.

SQINV and INV. The cost-sensitive re-weighting methods can be used for imbalanced regression by first dividing the target space into a finite number of bins. By calculating the number of samples in each bin, standard imbalanced classification method can be adopted. To fairly compare with state-of-the-art experiments, we also use square-root inverse-frequency weighting (SQINV) for *IMDB-WIKI-DIR* and *AgeDB-DIR*, and inverse-frequency weighting (INV) for *STS-B-DIR*.

LDS. Label Distribution Smoothing (Yang et al., 2021) convolves a symmetric kernel with the empirical label distribution to account for the continuity of labels. It has two hyperparameters—the kernel size and the standard deviation. In all experiments with LDS in this paper, we use Gaussian kernel with the validated hyperparameters in the original papers (the kernel size as 5 and the standard deviation as 2).

FDS. Feature Distribution Smoothing (Yang et al., 2021) performs distribution smoothing on the feature space. It transfers the feature statistics between nearby target bins. Similar to LDS, it has two hyperparameters—the kernel size and the standard deviation. In all experiments with FDS in this paper, we use Gaussian kernel with the validated hyperparameters in the original papers (the kernel size as 5 and the standard deviation as 2).

B. Experiment Details

In this section, we provide the full details about training details and additional experimental results. We train all models with 4 GPUs. We directly use all standard train/val/test splits created by Yang et al. (2021) for fair comparison. The details about training are provided in the subsections below.

B.1. Full Evaluation Metrics

In the main manuscript, we report MAE and GM for *IMDB-WIKI-DIR* and *AgeDB-DIR*, and report MSE and Pearson correlation for *STS-B-DIR*. We provide more metrics to evaluate RankSim and other baselines.

MAE. Mean absolute error (MAE) is defined as $\frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$. For i -th sample, y_i is the ground truth label, \hat{y}_i is the prediction, and N is the number of samples. Lower is better.

MSE. Mean squared error (MSE) is defined as $\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$. For i -th sample, y_i is the ground truth label, \hat{y}_i is the prediction, and N is the number of samples. Lower is better.

GM. Geometric Mean (GM) is defined as $(\prod_{i=1}^N |y_i - \hat{y}_i|)^{\frac{1}{N}}$. For i -th sample, y_i is the ground truth label, \hat{y}_i is the prediction, and N is the number of samples. Lower is better.

Pearson correlation. Following the original evaluation on *STS-B*, we use Pearson correlation, defined as $\frac{\text{cov}(y, \hat{y})}{\sigma_y \sigma_{\hat{y}}}$, to evaluate the linear relationship between predictions and ground truth labels. Please refer to the GLUE benchmark (Wang et al., 2018) for more details. Higher is better.

Spearman correlation. Following the original evaluation on *STS-B*, we use Spearman correlation, defined as the Pearson correlation between the rank variables, to evaluate the linear relationship between predictions and ground truth labels. Please refer to the GLUE benchmark (Wang et al., 2018) for more details. Higher is better.

B.2. Experiments on *IMDB-WIKI-DIR*

In Table 1, we report the experimental results on *IMDB-WIKI-DIR* with MAE and GM. In this section, we further report additional evaluation metrics.

Training details. For all experiments, we use ResNet-50 as the backbone network. We use Adam optimizer with 0.9 momentum and 1e-4 weight decay. The learning rate is 1e-3 and the batch size is 256. We train all methods for 90 epochs with The learning rate scheduled to drop by 10 times at epoch 60 and epoch 80. The input image size is 224 by 224. For RankSim hyperparameters, we set the balancing weight γ as 100 and the interpolation strength λ as 2 for all experiments on *IMDB-WIKI-DIR*. In all experiments, the best epoch is selected based on the ‘All’ MAE on the validation set.

Table 7. Complete Results on *IMDB-WIKI-DIR*. Baseline numbers are quoted from (Yang et al., 2021). The best results for each method (Vanilla, Focal-R, RRT, SQINV) are in **bold**. The best results for each metric and data subset (entire column) are in **bold and red**.

Metrics	MAE ↓				GM ↓				MSE ↓			
	All	Many	Med.	Few	All	Many	Med.	Few	All	Many	Med.	Few
VANILLA	8.06	7.23	15.12	26.33	4.57	4.17	10.59	20.46	138.06	108.70	366.09	964.92
VANILLA + LDS	7.83	7.31	12.43	22.51	4.42	4.19	7.00	13.94	131.65	109.04	298.98	829.35
VANILLA + FDS	7.85	7.18	13.35	24.12	4.47	4.18	8.18	15.18	133.81	107.51	332.90	916.18
VANILLA + LDS + FDS	7.78	7.20	12.61	22.19	4.37	4.12	7.39	12.61	129.35	106.52	311.49	811.82
VANILLA + RANKSIM	7.72	6.93	14.48	25.38	4.27	3.90	10.02	17.84	130.60	102.88	343.01	934.21
VANILLA + LDS + RANKSIM	7.57	7.00	12.16	22.44	4.23	4.00	6.81	13.23	127.29	104.07	299.81	844.00
VANILLA + FDS + RANKSIM	7.74	6.93	14.71	24.91	4.34	3.96	10.35	16.85	129.78	101.21	350.91	940.71
VANILLA + LDS + FDS + RANKSIM	7.69	7.13	12.30	21.43	4.34	4.13	6.72	12.48	129.09	106.21	303.82	798.05
FOCAL-R	7.97	7.12	15.14	26.96	4.49	4.10	10.37	21.20	136.98	106.87	368.60	1002.90
FOCAL-R + LDS	7.90	7.10	14.72	25.84	4.47	4.09	10.11	19.14	132.81	105.62	354.37	949.03
FOCAL-R + FDS	7.96	7.14	14.71	26.06	4.51	4.12	10.16	19.56	133.74	105.35	351.00	958.91
FOCAL-R + LDS + FDS	7.88	7.10	14.08	25.75	4.47	4.11	9.32	18.67	132.58	105.33	338.65	944.92
FOCAL-R + RANKSIM	7.77	6.99	14.23	26.01	4.38	4.03	9.25	20.16	130.86	102.82	342.70	967.78
FOCAL-R + LDS + RANKSIM	7.71	6.99	13.65	24.97	4.31	3.98	8.72	17.56	131.63	104.88	333.44	932.95
FOCAL-R + FDS + RANKSIM	7.75	7.01	14.06	24.56	4.33	3.99	9.04	16.26	132.42	105.45	338.61	918.23
FOCAL-R + LDS + FDS + RANKSIM	7.67	6.91	14.07	25.01	4.28	3.93	9.38	18.41	130.20	102.94	338.73	923.43
RRT	7.81	7.07	14.06	25.13	4.35	4.03	8.91	16.96	132.99	105.73	341.36	928.26
RRT + LDS	7.79	7.08	13.76	24.64	4.34	4.02	8.72	16.92	132.91	105.97	338.98	916.98
RRT + FDS	7.65	7.02	12.68	23.85	4.31	4.03	7.58	16.28	129.88	104.63	310.69	890.04
RRT + LDS + FDS	7.65	7.06	12.41	23.51	4.31	4.07	7.17	15.44	129.14	105.92	306.69	880.13
RRT + RANKSIM	7.55	6.83	13.47	24.72	4.17	3.86	8.66	15.54	127.78	101.16	325.96	945.92
RRT + LDS + RANKSIM	7.56	6.83	13.06	24.78	4.23	3.91	8.55	17.44	127.13	100.05	329.27	954.99
RRT + FDS + RANKSIM	7.35	6.81	11.50	22.75	4.05	3.85	6.05	14.68	123.18	100.86	280.55	879.85
RRT + LDS + FDS + RANKSIM	7.37	6.80	11.83	23.11	4.06	3.84	6.33	14.71	123.61	100.64	287.16	889.62
SQINV	7.87	7.24	12.44	22.76	4.47	4.22	7.25	15.10	134.36	111.23	308.63	834.08
SQINV + LDS	7.83	7.31	12.43	22.51	4.42	4.19	7.00	13.94	131.65	109.04	298.98	829.35
SQINV + FDS	7.83	7.23	12.60	22.37	4.42	4.20	6.93	13.48	132.64	109.28	311.35	851.06
SQINV + LDS + FDS	7.78	7.20	12.61	22.19	4.37	4.12	7.39	12.61	129.35	106.52	311.49	811.82
SQINV + RANKSIM	7.42	6.84	12.12	22.13	4.10	3.87	6.74	12.78	123.76	101.02	296.73	793.55
SQINV + LDS + RANKSIM	7.57	7.00	12.16	22.44	4.23	4.00	6.81	13.23	127.29	104.07	299.81	844.00
SQINV + FDS + RANKSIM	7.50	6.93	12.09	21.68	4.19	3.97	6.65	13.28	125.30	102.68	299.10	777.48
SQINV + LDS + FDS + RANKSIM	7.69	7.13	12.30	21.43	4.34	4.13	6.72	12.48	129.09	106.21	303.82	798.05

B.3. Experiments on *AgeDB-DIR*

Training details In the main manuscript, we reported RankSim results trained with batch size of 64 and learning rate of $2.5e-4$, and quoted the baseline numbers from the original paper (Yang et al., 2021). We found that this batch size and learning rate can also often produce better baseline results. To provide a full comparison, we include these additional baseline results in Table 8 in brackets. RankSim outperforms the baselines in both hyperparameter settings. We train all methods for 90 epochs with Adam optimizer. The momentum is 0.9 and the weight decay is $1e-4$. The learning rate scheduled to drop by 10 times at epoch 60 and epoch 80. The input image size is 224 by 224. For RankSim hyperparameters, we set the balancing weight γ as 100 and the interpolation strength λ as 2 for all experiments (for both batch size of 64 and 256) on *AgeDB-DIR*. In all experiments, the best epoch is selected based on the ‘All’ MAE on the validation set.

Table 8. Complete Results on *AgeDB-DIR*. The best results for each method (Vanilla, Focal-R, RRT, SQINV) are in **bold**. The best results for each metric and data subset (entire column) are in **bold and red**. We use batch size of 64 and learning rate of $2.5e-4$ for RankSim. Baseline numbers are quoted from (Yang et al., 2021). We find in some experiments baselines can perform better in the setting of batch size of 64. Thus we include the results with batch size of 64 in brackets for fair comparison.

Metrics	MAE ↓				GM ↓				MSE ↓			
	All	Many	Med.	Few	All	Many	Med.	Few	All	Many	Med.	Few
VANILLA	7.77 (7.35)	6.62 (6.56)	9.55 (8.23)	13.67 (12.37)	5.05 (4.82)	4.23 (4.32)	7.01 (5.49)	10.75 (9.47)	101.60 (93.47)	78.40 (73.27)	138.52 (121.67)	253.74 (241.97)
VANILLA + LDS	7.67 (7.39)	6.98 (6.72)	8.86 (8.18)	10.89 (10.25)	4.85 (4.65)	4.39 (4.30)	5.80 (5.25)	7.45 (6.91)	102.22 (91.79)	83.62 (77.24)	128.73 (111.52)	204.64 (174.73)
VANILLA + FDS	7.55 (7.42)	6.50 (6.46)	8.97 (8.64)	13.01 (13.07)	4.75 (4.77)	4.03 (4.13)	6.42 (5.96)	9.93 (10.27)	98.55 (93.42)	75.06 (70.29)	123.58 (118.25)	235.70 (243.12)
VANILLA + LDS + FDS	7.55 (7.44)	7.01 (6.85)	8.24 (8.36)	10.79 (10.43)	4.72 (4.79)	4.36 (4.34)	5.45 (5.79)	6.79 (7.15)	99.46 (92.38)	84.10 (78.99)	112.20 (107.32)	209.27 (177.45)
VANILLA + RANKSIM	7.13	6.51	8.17	10.12	4.48	4.01	5.27	6.79	87.45	71.84	111.41	168.61
VANILLA + LDS + RANKSIM	6.99	6.38	7.88	10.23	4.40	3.97	5.30	6.93	84.14	71.72	98.59	161.48
VANILLA + FDS + RANKSIM	7.33	6.49	8.53	11.98	4.82	4.19	6.16	8.99	90.09	69.90	112.69	205.12
VANILLA + LDS + FDS + RANKSIM	7.03	6.54	7.68	9.92	4.45	4.07	5.23	6.35	84.96	74.27	93.64	161.92
FOCAL-R	7.64 (7.41)	6.68 (6.63)	9.22 (8.54)	13.00 (11.68)	4.90 (4.72)	4.26 (4.28)	6.39 (5.36)	9.52 (8.39)	101.26 (94.08)	77.03 (74.99)	131.81 (116.39)	252.47 (212.67)
FOCAL-R + LDS	7.56 (7.25)	6.67 (6.41)	8.82 (8.48)	12.40 (11.77)	4.82 (4.68)	4.27 (4.05)	5.87 (5.74)	8.83 (7.84)	98.80 (90.15)	77.14 (70.01)	125.53 (112.42)	229.36 (216.84)
FOCAL-R + FDS	7.65 (7.33)	6.89 (6.51)	8.70 (8.49)	11.92 (11.97)	4.83 (4.78)	4.32 (4.23)	5.89 (5.79)	8.04 (8.00)	100.41 (90.73)	80.97 (71.64)	121.84 (113.45)	208.25 (208.17)
FOCAL-R + LDS + FDS	7.47 (7.32)	6.69 (6.60)	8.99 (8.26)	12.83 (11.53)	4.93 (4.75)	4.27 (4.26)	6.25 (5.61)	9.98 (8.45)	96.51 (90.91)	75.02 (74.23)	123.43 (108.77)	225.04 (199.10)
FOCAL-R + RANKSIM	7.15	6.45	7.97	11.50	4.53	4.10	5.10	8.50	86.67	69.57	105.10	197.15
FOCAL-R + LDS + RANKSIM	7.25	6.40	8.71	11.24	4.58	4.02	5.99	7.52	89.54	69.87	118.84	194.34
FOCAL-R + FDS + RANKSIM	7.25	6.72	7.86	10.58	4.54	4.22	4.84	7.57	90.12	76.69	105.55	175.18
FOCAL-R + LDS + FDS + RANKSIM	7.09	6.17	8.71	11.68	4.46	3.85	5.76	8.78	87.12	66.01	120.10	195.35
RRT	7.74 (7.29)	6.98 (6.68)	8.79 (7.89)	11.99 (11.36)	5.00 (4.61)	4.50 (4.24)	5.88 (5.01)	8.63 (8.13)	102.89 (91.30)	83.37 (75.21)	125.66 (108.59)	224.04 (195.38)
RRT + LDS	7.72 (7.52)	7.00 (6.86)	8.75 (8.22)	11.62 (11.82)	4.98 (4.90)	4.54 (4.42)	5.71 (5.66)	8.27 (8.63)	102.63 (95.85)	83.93 (79.85)	126.01 (110.95)	214.66 (205.24)
RRT + FDS	7.70 (7.30)	6.95 (6.64)	8.76 (8.01)	11.86 (11.44)	4.82 (4.62)	4.32 (4.24)	5.83 (5.06)	8.08 (8.09)	102.09 (91.60)	84.49 (74.79)	122.89 (110.78)	224.05 (197.30)
RRT + LDS + FDS	7.66 (7.29)	6.99 (6.70)	8.60 (7.85)	11.32 (11.26)	4.80 (4.62)	4.42 (4.28)	5.53 (4.95)	6.99 (7.75)	101.74 (91.35)	83.12 (75.93)	121.08 (107.46)	210.78 (192.41)
RRT + RANKSIM	7.11	6.53	8.00	10.04	4.52	4.19	5.05	6.77	86.93	72.11	108.08	168.41
RRT + LDS + RANKSIM	6.94	6.43	7.54	10.10	4.37	3.97	5.11	7.05	82.98	72.49	91.24	159.24
RRT + FDS + RANKSIM	7.11	6.55	7.99	10.02	4.49	4.13	5.13	6.85	86.93	72.36	107.90	166.55
RRT + LDS + FDS + RANKSIM	7.13	6.54	8.07	10.12	4.55	4.18	5.20	6.87	87.28	72.20	109.19	169.16
SQINV	7.81 (7.49)	7.16 (6.92)	8.80 (8.34)	11.20 (10.55)	4.99 (4.83)	4.57 (4.53)	5.73 (5.25)	7.77 (7.05)	105.14 (93.89)	87.21 (77.68)	127.66 (117.02)	212.30 (183.03)
SQINV + LDS	7.67 (7.42)	6.98 (6.83)	8.86 (8.21)	10.89 (10.79)	4.85 (4.73)	4.39 (4.37)	5.80 (5.40)	7.45 (7.03)	102.22 (95.10)	83.62 (78.63)	128.73 (111.75)	204.64 (204.60)
SQINV + FDS	7.69 (7.55)	7.10 (6.99)	8.86 (8.40)	10.89 (10.48)	4.85 (4.82)	4.39 (4.49)	5.80 (5.47)	7.45 (6.58)	102.22 (97.35)	83.62 (81.90)	128.73 (118.30)	204.64 (155.32)
SQINV + LDS + FDS	7.55 (7.37)	7.01 (6.82)	8.24 (8.25)	10.79 (10.16)	4.72 (4.72)	4.36 (4.33)	5.45 (5.50)	6.79 (6.98)	99.46 (93.45)	84.10 (79.86)	112.20 (110.00)	209.27 (184.41)
SQINV + RANKSIM	6.91	6.34	7.79	9.89	4.28	3.92	4.88	6.89	82.10	68.60	102.61	152.84
SQINV + LDS + RANKSIM	6.99	6.38	7.88	10.23	4.40	3.97	5.30	6.90	84.14	71.72	98.59	161.48
SQINV + FDS + RANKSIM	7.02	6.49	7.84	9.68	4.53	4.13	5.37	6.89	83.51	71.99	99.14	149.05
SQINV + LDS + FDS + RANKSIM	7.03	6.54	7.68	9.92	4.45	4.07	5.23	6.35	84.96	74.27	93.64	161.92

B.4. Experiments on *STS-B-DIR*

Training details We follow the same training details as Yang et al. (2021)—train with Adam optimizer, validate the model every 10 epochs, use MSE as the validation metric, and stop training when the validation error does not decrease after 10 validation checks. We use a smaller batch size of 16 and learning rate of $2.5e-4$ as these produce better results (lower ‘All’ MSE error) on the validation set. We set the maximum number of validation checks as 300, and patience for early stopping as 30. For RankSim hyperparameters, we set the balancing weight γ as $3e-4$ and the interpolation strength λ as 2 for all experiments on *STS-B-DIR*.

C. Ablation Studies and Analysis

C.1. Different loss functions for ranking similarity

We provide full details of the ablation study on the ranking similarity loss function ℓ in Eq. 4. We consider five loss functions—mean squared error (MSE), mean absolute error (MAE), cosine distance, huber loss, and L_∞ loss. Please refer to the main manuscript for the mathematical formulation of each function. In all other experiments in this paper, we keep

Table 9. Complete Results on *STS-B-DIR*. The best results for each method (Vanilla, Focal-R, RRT, SQINV) are in **bold**. The best results for each metric and data subset (entire column) are in **bold and red**. For RankSim, we use batch size of 16 and learning rate of 2.5e-4. Baseline numbers are quoted from (Yang et al., 2021).

Metrics	MSE ↓				MAE ↓				Pearson correlation (%) ↑				Spearman correlation (%) ↑			
	All	Many	Med.	Few	All	Many	Med.	Few	All	Many	Med.	Few	All	Many	Med.	Few
VANILLA	0.974	0.851	1.520	0.984	0.794	0.740	1.043	0.771	74.2	72.0	62.7	75.2	74.4	68.8	50.5	75.0
VANILLA + LDS	0.914	0.819	1.319	0.955	0.773	0.729	0.970	0.772	75.6	73.4	63.8	76.2	76.1	70.4	55.6	74.3
VANILLA + FDS	0.916	0.875	1.027	1.086	0.767	0.746	0.840	0.811	75.5	73.0	67.0	72.8	75.8	69.9	54.4	72.0
VANILLA + LDS + FDS	0.907	0.802	1.363	0.942	0.766	0.718	0.986	0.755	76.0	74.0	65.2	76.6	76.4	70.7	54.9	74.9
VANILLA + RANKSIM	0.873	0.908	0.767	0.705	0.749	0.755	0.737	0.695	76.8	71.0	72.9	85.2	77.2	68.3	55.4	88.4
VANILLA + LDS + RANKSIM	0.889	0.911	0.849	0.690	0.755	0.762	0.758	0.638	76.2	70.7	70.0	85.6	76.3	67.8	49.0	85.4
VANILLA + FDS + RANKSIM	0.884	0.924	0.767	0.685	0.755	0.769	0.736	0.653	76.5	70.4	72.5	85.7	76.7	67.1	53.5	87.8
VANILLA + LDS + FDS + RANKSIM	0.903	0.908	0.911	0.804	0.761	0.759	0.786	0.712	75.8	70.6	69.0	82.7	75.8	67.3	49.3	84.9
FOCAL-R	0.951	0.843	1.425	0.957	0.790	0.739	1.028	0.759	74.6	72.3	61.8	76.4	75.0	69.4	51.9	75.5
FOCAL-R + LDS	0.930	0.807	1.449	0.993	0.781	0.723	1.031	0.801	75.7	73.9	62.4	75.4	76.2	71.2	50.7	74.7
FOCAL-R + FDS	0.920	0.855	1.169	1.008	0.775	0.743	0.903	0.804	75.1	72.6	66.4	74.7	75.4	69.4	52.7	75.4
FOCAL-R + LDS + FDS	0.940	0.849	1.358	0.916	0.785	0.737	0.984	0.732	74.9	72.2	66.3	77.3	75.1	69.2	52.5	76.4
FOCAL-R + RANKSIM	0.887	0.889	0.918	0.745	0.763	0.757	0.805	0.719	76.2	70.8	70.4	84.6	76.7	68.2	51.0	87.7
FOCAL-R + LDS + RANKSIM	0.872	0.887	0.847	0.718	0.752	0.751	0.770	0.701	76.7	71.2	70.3	85.1	77.1	68.2	50.1	88.4
FOCAL-R + FDS + RANKSIM	0.913	0.952	0.793	0.723	0.763	0.776	0.735	0.660	75.6	69.6	71.5	84.8	75.8	66.9	49.7	87.7
FOCAL-R + LDS + FDS + RANKSIM	0.911	0.943	0.779	0.866	0.767	0.765	0.725	0.747	75.7	69.9	71.4	81.2	75.7	67.0	47.7	80.7
RRT	0.964	0.842	1.503	0.978	0.793	0.739	1.044	0.768	74.5	72.4	62.3	75.4	74.7	69.2	51.3	74.7
RRT + LDS	0.916	0.817	1.344	0.945	0.772	0.727	0.980	0.756	75.7	73.5	64.1	76.6	76.1	70.4	53.2	74.2
RRT + FDS	0.929	0.857	1.209	1.025	0.769	0.736	0.905	0.795	74.9	72.1	67.2	74.0	75.0	69.1	52.8	74.6
RRT + LDS + FDS	0.903	0.806	1.323	0.936	0.764	0.719	0.965	0.760	76.0	73.8	65.2	76.7	76.4	70.8	54.7	74.7
RRT + RANKSIM	0.865	0.876	0.867	0.670	0.748	0.749	0.767	0.670	77.1	72.2	68.3	86.1	77.4	69.6	48.0	89.4
RRT + LDS + RANKSIM	0.874	0.893	0.833	0.722	0.754	0.758	0.752	0.698	77.0	72.3	68.3	84.8	77.2	69.5	47.2	87.8
RRT + FDS + RANKSIM	0.871	0.874	0.898	0.734	0.750	0.748	0.779	0.687	76.8	72.0	68.7	84.5	77.0	69.4	47.1	87.2
RRT + LDS + FDS + RANKSIM	0.882	0.892	0.887	0.702	0.758	0.759	0.775	0.681	76.6	71.7	68.0	85.5	76.8	69.0	46.5	88.3
INV	1.005	0.894	1.482	1.046	0.805	0.761	1.016	0.780	72.8	70.3	62.5	73.2	73.1	67.2	54.1	71.4
INV + LDS	0.914	0.819	1.319	0.955	0.773	0.729	0.970	0.772	75.6	73.4	63.8	76.2	76.1	70.4	55.6	74.3
INV + FDS	0.927	0.851	1.225	1.012	0.771	0.740	0.914	0.756	75.0	72.4	66.6	74.2	75.2	69.2	55.2	74.8
INV + LDS + FDS	0.907	0.802	1.363	0.942	0.766	0.718	0.986	0.755	76.0	74.0	65.2	76.6	76.4	70.7	54.9	74.9
INV + RANKSIM	1.091	1.056	1.240	1.118	0.854	0.843	0.912	0.822	69.9	65.2	60.1	76.0	70.2	62.5	45.0	78.5
INV + LDS + RANKSIM	0.889	0.911	0.849	0.690	0.755	0.762	0.758	0.638	76.2	70.7	70.0	85.6	76.3	67.8	49.0	85.4
INV + FDS + RANKSIM	1.083	1.035	1.301	1.063	0.831	0.812	0.914	0.840	70.0	64.8	68.9	76.7	69.7	61.6	43.8	82.5
INV + LDS + FDS + RANKSIM	0.903	0.908	0.911	0.804	0.761	0.759	0.786	0.712	75.8	70.6	69.0	82.7	75.8	67.3	49.3	84.9

this ranking similarity loss function as MSE. We conduct experiments on AgeDB-DIR dataset and choose ResNet-50 as backbone network. Beside the cosine similarity we used in other experiments, in this experiment we finetune RankSim with other ranking similarity loss functions. For the hyperparameters of RankSim, we set cosine distance, γ as 1 and λ as 2; Huber loss function: γ as 1 and λ as 2; L_∞ loss: γ as 500 and λ as 2; MAE loss: γ as 200 and λ as 2.

C.2. Different similarity functions for feature similarity

We provide full details of the ablation study on the feature similarity function σ in Eq. 3. We consider five similarity functions—cosine similarity, correlation similarity, MSE similarity, MAE similarity, and L_∞ similarity. As this function is a similarity function, for MSE/MAE/ L_∞ similarity, we simply negate the loss function of MSE/MAE/ L_∞ to be its similarity function. Please refer to the main manuscript for the mathematical formulation of each function. In all other experiments in this paper, we keep this feature similarity function as cosine similarity. We conduct experiments on AgeDB-DIR dataset and choose ResNet-50 as backbone network. Beside the cosine similarity we used in other experiments, in this experiment we finetune RankSim with other feature similarity functions. For the hyperparameters of RankSim, we set correlation similarity function, γ as 0.1 and λ as 2; MSE similarity function: γ as 50 and λ as 2; MAE similarity function: γ as 200 and λ as 2; L_∞ similarity function: γ as 500 and λ as 2.

C.3. Qualitative visualization of ranking matrices

The visualized matrices are obtained by applying the ranking operation \mathbf{rk} on each row of \mathcal{S}^y and \mathcal{S}^z as in Eq. 4, and averaging the result over all test batches. For example, row 1 in the matrix indicates the averaged rankings with respect to the first (youngest) data sample in a batch. Suppose we have a toy test set consisting of the batch in Fig. 1 ($y = [1, 21, 25, 70]$ and $\mathbf{rk}(\mathcal{S}_{[1,:]}^y) = [1, 3, 4, 2]$), plus one additional batch in which $y = [5, 10, 40, 56]$ and $\mathbf{rk}(\mathcal{S}_{[1,:]}^z) = [1, 2, 4, 3]$. Then row 1 of the average feature-space ranking matrix would be given by the average of $[1, 3, 4, 2]$ and $[1, 2, 4, 3]$: that is, $[1, 2.5, 4, 2.5]$.

C.4. Sensitivity to hyperparameters

In this subsection, we present sensitivity experiments on the two hyperparameters of RankSim (the balancing weight γ and the interpolation strength λ) on *AgeDB-DIR* in Table 10 and *IMDB-WIKI-DIR* in Table 11.

Table 10. Varying hyperparameters γ and λ . We conduct experiments on *AgeDB-DIR* with SQINV. In the first section, we keep λ as 2 and change the value of γ (the best result is in **bold**). In the second section, we keep γ as 100 and change the value of λ (the best result is in **bold**). The best result for each metric in all experiments (entire column) is in **bold and red**.

hyperparams		MAE ↓				GM ↓				MSE ↓			
γ	λ	All	Many	Med.	Few	All	Many	Med.	Few	All	Many	Med.	Few
0.01	2	7.42	6.75	8.42	11.01	4.76	4.31	5.63	7.59	94.68	77.19	114.39	205.43
0.1	2	7.36	6.72	8.44	10.42	4.72	4.31	5.58	7.00	91.86	76.50	114.43	174.48
1	2	7.17	6.56	8.23	10.08	4.49	4.10	5.30	6.78	89.20	75.28	109.53	164.49
10	2	7.05	6.55	7.79	9.72	4.56	4.22	5.18	6.53	84.57	72.71	99.37	155.74
100	2	6.91	6.34	7.79	9.89	4.28	3.92	4.88	6.89	82.10	68.60	102.61	152.84
200	2	7.15	6.52	8.31	9.94	4.54	4.10	5.67	6.45	87.45	73.51	108.96	159.55
500	2	7.18	6.54	8.17	10.46	4.55	4.12	5.45	7.01	88.86	73.57	110.19	174.29
1000	2	7.17	6.47	8.44	10.26	4.54	4.13	5.46	6.80	89.71	72.37	119.44	171.34
2000	2	7.10	6.50	7.99	10.27	4.50	4.08	5.35	7.06	87.23	72.98	103.97	175.56
3000	2	7.14	6.70	7.73	9.66	4.61	4.33	4.99	6.72	86.76	75.35	102.49	151.02
100	0.05	7.76	7.02	9.25	10.67	4.97	4.53	6.06	6.92	103.20	83.02	140.37	191.01
100	0.1	7.32	6.72	8.21	10.55	4.65	4.25	5.47	6.98	91.37	76.59	110.95	176.76
100	0.5	7.11	6.64	7.70	9.93	4.60	4.28	5.09	6.96	86.61	74.46	99.88	164.67
100	1	7.26	6.63	8.31	10.24	4.63	4.18	5.64	6.93	90.42	76.13	113.52	161.46
100	1.5	7.07	6.45	8.22	9.79	4.47	4.06	5.40	6.66	85.11	71.15	107.27	155.61
100	2	6.91	6.34	7.79	9.89	4.28	3.92	4.88	6.89	82.10	68.60	102.61	152.84
100	4	7.14	6.60	7.84	10.31	4.55	4.25	5.03	6.69	87.31	73.84	102.94	171.33
100	8	7.26	6.85	7.82	9.61	4.68	4.42	5.10	6.36	89.91	79.06	104.37	152.39
100	16	7.14	6.66	7.76	9.91	4.56	4.27	4.99	6.56	87.02	74.28	104.00	160.32
100	32	7.25	6.64	8.28	10.18	4.78	4.35	5.77	7.02	87.77	73.31	107.10	170.77
100	64	7.11	6.51	7.90	10.69	4.55	4.10	5.47	7.19	85.06	71.60	96.55	180.44

Table 11. Varying hyperparameters γ and λ . We conduct experiments on *IMDB-WIKI-DIR* with SQINV. In the first section, we keep λ as 2 and change the value of γ (the best result is in **bold**). In the second section, we keep γ as 100 and change the value of λ (the best result is in **bold**). The best result for each metric in all experiments (entire column) is in **bold and red**.

hyperparams		MAE ↓				GM ↓				MSE ↓			
γ	λ	All	Many	Med.	Few	All	Many	Med.	Few	All	Many	Med.	Few
0.01	2	7.67	7.08	12.42	22.86	4.32	4.07	7.25	13.52	129.26	105.88	302.88	851.47
0.1	2	7.61	6.99	12.67	23.31	4.26	4.01	7.19	13.34	127.55	102.95	311.63	876.28
1	2	7.47	6.88	12.25	22.89	4.17	3.93	7.02	14.08	125.15	101.97	295.09	859.42
10	2	7.43	6.83	12.22	23.19	4.16	3.93	6.80	14.50	123.81	99.98	300.47	862.53
100	2	7.42	6.84	12.12	22.13	4.10	3.87	6.74	12.78	123.76	101.02	296.73	793.55
200	2	7.44	6.88	11.77	22.75	4.13	3.91	6.53	14.93	124.68	102.11	290.17	838.68
500	2	7.43	6.87	11.96	21.76	4.14	3.92	6.61	12.89	123.78	101.25	295.44	785.79
1000	2	7.41	6.83	12.13	21.54	4.14	3.92	6.78	13.16	122.67	99.85	299.33	769.95
2000	2	7.58	6.98	12.35	22.81	4.22	3.98	6.89	15.13	127.88	103.97	306.62	856.13
3000	2	7.48	6.91	11.95	22.65	4.15	3.94	6.35	14.14	126.39	103.21	298.52	842.27
100	0.05	7.41	6.87	11.72	22.22	4.15	3.92	6.67	14.57	124.49	102.49	284.39	831.26
100	0.1	7.41	6.84	11.89	22.34	4.11	3.89	6.60	13.32	124.32	101.87	289.18	833.23
100	0.5	7.37	6.82	11.76	22.05	4.05	3.83	6.43	12.71	124.36	101.88	289.72	831.52
100	1	7.50	6.94	11.98	22.49	4.23	4.00	6.78	14.19	124.39	102.01	287.93	836.41
100	1.5	7.50	6.95	11.88	22.39	4.21	3.98	6.85	14.33	126.34	104.34	287.73	821.28
100	2	7.42	6.84	12.12	22.13	4.10	3.87	6.74	12.78	123.76	101.02	296.73	793.55
100	4	7.60	7.03	12.19	22.78	4.28	4.06	6.83	14.45	128.59	105.64	297.42	850.14
100	8	7.53	6.93	12.33	23.23	4.24	4.00	7.09	15.37	125.32	101.61	300.69	863.02
100	16	7.53	6.93	12.38	23.23	4.22	3.97	7.02	15.02	126.81	103.04	302.43	867.96
100	32	7.62	7.09	11.93	21.51	4.33	4.12	6.64	12.58	127.07	105.14	290.92	796.45
100	64	7.65	7.09	12.05	23.02	4.31	4.08	6.78	15.41	128.65	106.24	292.17	844.65

C.5. Varying batch size

Since RankSim requires a batch-wise calculation to construct the pairwise similarity matrices, we also conducted experiments in which we varied the batch size.

Table 12. Varying the batch size. Results are on *AgeDB-DIR* with SQINV re-weighting. We keep λ as 2 and γ as 100.

Metrics	MAE ↓				GM ↓				MSE ↓			
Shot	All	Many	Med.	Few	All	Many	Med.	Few	All	Many	Med.	Few
VANILLA	7.77	6.62	9.55	13.67	5.05	4.23	7.01	10.75	101.60	78.40	138.52	253.74
SQINV + LDS + FDS	7.55	7.01	8.24	10.79	4.72	4.36	5.45	6.79	99.46	84.10	112.20	209.27
SQINV + RANKSIM, batch size 64	6.91	6.34	7.79	9.89	4.28	3.92	4.88	6.89	82.10	68.60	102.61	152.84
SQINV + RANKSIM, batch size 128	7.13	6.58	8.07	9.79	4.40	3.95	5.38	7.02	88.93	76.68	106.52	156.00
SQINV + RANKSIM, batch size 256	7.17	6.63	7.98	10.06	4.53	4.18	5.24	6.38	89.37	75.73	105.36	173.92