Semi-supervised Regression by Preserving Ranking Relationships between Close Unlabeled Samples

Ximing Li[1,2,3], **Jiaxuan Jiang**[1,2], **Changchun Li**[1,2*], **You Lu**[4], **Renchu Guan**[1,2]
[1]College of Computer Science and Technology, Jilin University, China
[2]Key Laboratory of Symbolic Computation and Knowledge Engineering of the Ministry of Education, Jilin University, China
[3]RIKEN Center for Advanced Intelligence Project, Japan
[4]Artificial Intelligence Research Institute, Shenzhen University of Advanced Technology, China
{liximing86, jiaxuanjiang.jxj, changchunli93}@gmail.com, luyou@suat-sz.edu.cn, guanrenchu@jlu.edu.cn

Abstract

Semi-**S**upervised **L**earning (**SSL**) aims to improve the learning performance of supervised learning with a large number of unlabeled samples. Existing SSL methods such as FixMatch and FlexMatch select unlabeled samples with high-confidence pseudo-labels and make consistency constraints between their weak and strong augmentations. Unfortunately, they cannot be applied **S**emi-**S**upervised **R**egression (**SSR**) because regression predictions can not reflect the confidence of pseudo-labels. To solve this, a recent SSR method RankUp incorporates an auxiliary ranking task by leveraging sample pairs with high-confidence pseudo-ranks. In this paper, we upgrade RankUp to a novel SSR method, namely **S**emi-**S**upervised **R**egression by **R**anking **C**lose **U**nlabeled **S**amples (**SSR-RCUS**). Its basic idea is reconstructing closed mixup augmented samples with high-confident pseudo-ranks under a monotonicity assumption, and then applying them to the auxiliary ranking task to improve regression performance. We conduct extensive experiments to evaluate the performance of SSR-RCUS on benchmark datasets, and empirical results demonstrate that SSR-RCUS can outperform the existing baselines in various settings, especially when labeled data are scarce.

Code — https://github.com/changchunli/SSR-RCUS

Introduction

Deep learning methods have achieved promising success in many real-world applications (Dong, Wang, and Abbas 2021), especially those that depend on supervised learning paradigms containing a large amount of precisely labeled data. Unfortunately, in many specific scenarios, collecting high-quality labeled samples can be very costly; in parallel, acquiring abundant unlabeled samples can be much easier. Consequently, the community remains turning to the active topic of **S**emi-**S**upervised **L**earning (**SSL**) (vanEngelen and Hoos 2020; Yang et al. 2023; Li et al. 2024a,b), whose objective is to improve the learning performance of supervised learning with a large number of unlabeled samples.

While there is a long history of SSL (Chapelle, Schölkopf, and Zien 2006), its key idea remains how to effectively leverage abundant unlabeled samples. Representative methodologies mainly include pseudo-labeling (Beyer et al. 2019; Xie et al. 2020b; Li, Li, and Ouyang 2021; Kou et al. 2025a,b), consistency regularization (Miyato et al. 2019; Xie et al. 2020a), graph-based methods (Wu et al. 2019; Pan et al. 2020), and hybrid methods (Verma et al. 2019a; Berthelot et al. 2019; Sohn et al. 2020; Zhang et al. 2021a), *etc.* In particular, some hybrid methods such as FixMatch (Sohn et al. 2020) and FlexMatch (Zhang et al. 2021a) select unlabeled samples with high-confident pseudo-labels by leveraging a (adaptive) confidence threshold, and make consistency constraints between pseudo-labels of their weak and strong augmentations. Empirically, they improve learning performance by a large margin and accordingly imply that **selecting unlabeled samples with high-confident pseudo-labels plays a key role in SSL**. These hybrid methods focus on classification problems, where one can directly measure pseudo-labels' confidences by the current predictions. However, they cannot be applied to regression problems mainly because regression predictions cannot reflect the confidence of pseudo-labels (Huang, Fu, and Tsao 2024).

To select unlabeled samples with high-confident pseudo-labels in **S**emi-**S**upervised **R**egression (**SSR**), a recent SSR method named RankUp proposes to leverage unlabeled samples by an auxiliary ranking task, which ingests unlabeled sample pairs and predicts the relative rank within each pair (Huang, Fu, and Tsao 2024). Accordingly, this ranking task's prediction, *i.e.* pseudo-rank, can directly measure the ranking confidence. For each unlabeled sample pair with high-confidence pseudo-rank, RankUp makes a consistency constraint between pseudo-ranks of their weak and strong augmentations. This strategy can preserve the relative rank between samples to improve regression performance. However, high-confident pseudo-ranks often happen for the sample pairs with a large gap between their response targets. Accordingly, we raise the question of whether we can preserve the relative rank between closed samples to improve regression performance, and the key problem becomes how to select closed samples with high-confidence pseudo-ranks.

To answer this question, we upgrade RankUp to a novel SSR method, namely **S**emi-**S**upervised **R**egression by **R**anking **C**lose **U**nlabeled **S**amples (**SSR-RCUS**). Its basic idea is reconstructing closed mixup augmented samples with high-confidence pseudo-ranks under a monotonicity assumption. Specifically, given any sample pair with a high-confidence pseudo-rank, we can generate mixup augmented

*Corresponding author.

samples with the pair. We make a monotonicity assumption for mixup augmented samples so the pseudo-ranks between any sample from the pair and mixup augmented samples are preserved. We can generate closed mixup augmented samples from the samples within the pair by controlling the interpolating coefficient of mixup, leading to closed samples with high-confident pseudo-ranks. They can then be applied to the auxiliary ranking task to improve regression performance. We conduct extensive experiments to evaluate the performance of SSR-RCUS on benchmark datasets, and empirical results demonstrate that SSR-RCUS can outperform the existing baselines in various settings, especially when labeled data are scarce.

Method

In this section, we describe the proposed SSR method named **SSR-RCUS**.

Preliminaries

Problem setup of SSR Consider a d-dimensional feature space $\mathcal{X} \subseteq \mathbb{R}^d$ and a regression target space $\mathcal{Y} \subseteq \mathbb{R}$. In the context of SSR, we have access to a labeled training dataset $\mathcal{D}_l = \{(\mathbf{x}_i^l, y_i^l)\}_{i=1}^{N_l}$ consisting of N_l samples, and an unlabeled training dataset $\mathcal{D}_u = \{\mathbf{x}_i^u\}_{i=1}^{N_u}$ containing N_u feature data only, where $\mathbf{x}_i^l, \mathbf{x}_i^u \in \mathcal{X}$ are the labeled and unlabeled feature vectors, respectively, and $y_i^l \in \mathcal{Y}$ is the regression target of $\mathbf{x}_i^l$. The goal of SSR is to induce a regression model $f : \mathcal{X} \to \mathcal{Y}$ from $\{\mathcal{D}_l, \mathcal{D}_u\}$. Generally, the regression model is designed as $f = r_{\mathbf{W}} \circ e_{\mathbf{\Theta}}$ with parameters $\{\mathbf{\Theta}, \mathbf{W}\}$, where $e_{\mathbf{\Theta}}$ is typically a deep encoder and $r_{\mathbf{W}}$ is a regressor. The deep encoder is used to transform the original feature $\mathbf{x}$ to a latent feature $\mathbf{z} = e_{\mathbf{\Theta}}(\mathbf{x})$. With $\mathbf{z}$, the regressor is then used to generate the prediction of the regression target $p = r_{\mathbf{W}}(\mathbf{z})$.

Generic objective of SSR Without concern for any specific SRR method, we introduce the generic case of SSR. To handle the training dataset $\{\mathcal{D}_l, \mathcal{D}_u\}$, the generic objective of SSR can be formulated with three parts including a supervised loss $\mathcal{L}_l$, an unsupervised loss $\mathcal{L}_u$, and regularization term(s) $\mathcal{R}$:

$$\mathcal{L}(\mathcal{D}_l,\mathcal{D}_u) = \mathcal{L}_l(\mathcal{D}_l) + \omega_u \mathcal{L}_u(\mathcal{D}_u) + \omega_r \mathcal{R} \tag{1}$$

where ω_u and ω_r are coefficient parameters. The supervised loss is the average loss concerning on $\mathcal{D}_l$, and it is generally specified as follows:

$$\mathcal{L}_l(\mathbf{\Theta}, \mathbf{W}; \mathcal{D}_l) = \frac{1}{N_l} \sum_{i=1}^{N_l} \ell_{reg}\big(p_i^l, y_i^l\big), \tag{2}$$

where ℓ_{reg} denotes a prevalent loss function used in regression such as mean squared error or mean absolute error. **The unsupervised loss refers to $\mathcal{D}_u$ and is the key idea for specific SSR methods**. Additionally, the regularization term(s) includes traditional norm-based constraints and specific SSL regularization such as entropy regularization (Grandvalet and Bengio 2004).

Mixup variants *Mixup* is an easy-to-implement data-agnostic augmentation methodology, and it follows a **linear correspondence assumption** that linear interpolations of features correspond to the same linear interpolations of associated targets (Zhang et al. 2018). Specifically, the standard mixup constructs a new augmented sample $(\widehat{\mathbf{x}}, \widehat{y})$ with any randomly drawn sample pair $\{(\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j)\}$:

$$\begin{aligned}\widehat{\mathbf{x}} =& \lambda \mathbf{x}_i + (1-\lambda)\mathbf{x}_j, \quad \widehat{y} = \lambda y_i + (1-\lambda) y_j, \\ & \lambda = \max(\lambda', 1-\lambda'),\ \lambda' \sim \mathrm{Beta}(\alpha, \alpha),\ \alpha \in (0, \infty).\end{aligned}$$

Manifold mixup extends the linear correspondence assumption to the latent feature space (Verma et al. 2019b), leading to the following latent augmented sample $(\widehat{\mathbf{z}}, \widehat{y})$:

$$\begin{aligned}\widehat{\mathbf{z}} =& \lambda \mathbf{z}_i + (1-\lambda)\mathbf{z}_j, \quad \widehat{y} = \lambda y_i + (1-\lambda) y_j, \\ & \lambda = \max(\lambda', 1-\lambda'),\ \lambda' \sim \mathrm{Beta}(\alpha, \alpha),\ \alpha \in (0, \infty).\end{aligned}$$

SSR-RCUS

Generally, our SSR-RCUS is built on the generic objective of SSR formulated in Eq.(1). Specifically, we specify the unsupervised loss with a ranking loss by preserving the relative rank among closed sample pairs with high-confident pseudo-ranks, which are constructed with the above easy-to-implement mixup technique.

The ranking loss preserves the relative rank between a sample pair by constructing a loss between the ranking prediction and target, which are obtained according to a ranking predictor and the sample pair's regression targets, respectively. Let $h_{\mathbf{H}}$ be a ranking predictor that ingests the latent features $(\mathbf{z}_i, \mathbf{z}_j)$ of a sample pair $((\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j))$ and outputs a ranking prediction $q_{ij} = h_{\mathbf{H}}(\mathbf{z}_i, \mathbf{z}_j)$, formally defined as follows:

$$q_{ij} = h_{\mathbf{H}}(\mathbf{z}_i, \mathbf{z}_j) = \frac{1}{1 + \exp(\mathbf{H}^\top \mathbf{z}_i - \mathbf{H}^\top \mathbf{z}_j)}. \tag{3}$$

Then, the ranking loss can be formulated as

$$\ell_r(q_{ij}, t_{ij}) = \ell_{ce}\big(q_{ij}, t_{ij}\big), \tag{4}$$

where ℓ_{ce} is the cross-entropy loss, $t_{ij} \in \{0, 1\}$ is the ranking target defined as $t_{ij} = \mathbb{I}(y_i > y_j)$, and $\mathbb{I}$ denotes the indicator function.

Rather than select sample pairs directly for the ranking task, to preserve the relative rank among closed sample pairs, we generate the corresponding mixup augmented samples close to samples within any interpolation pair by controlling the interpolation coefficient of mixup. Specifically, for any sample pair $((\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j))$, we generate two mixup augmented samples $\underline{\mathbf{x}}_{ij}$ and $\overline{\mathbf{x}}_{ij}$ by exchanging the interpolation coefficient as follows:

$$\begin{aligned}\underline{\mathbf{x}}_{ij} =& \lambda \mathbf{x}_i + (1-\lambda)\mathbf{x}_j, \quad \underline{y}_{ij} = \lambda y_i + (1-\lambda) y_j, \\ \overline{\mathbf{x}}_{ij} =& (1-\lambda)\mathbf{x}_i + \lambda \mathbf{x}_j, \quad \overline{y}_{ij} = (1-\lambda) y_i + \lambda y_j, \\ & \lambda = \max(\lambda', 1-\lambda'),\ \lambda' \sim \mathrm{Beta}(\alpha, \alpha),\ \alpha \in (0, \infty).\end{aligned}$$

Thus $\underline{\mathbf{x}}_{ij}$ is closer to $\mathbf{x}_i$, and $\overline{\mathbf{x}}_{ij}$ closer to $\mathbf{x}_j$. By exploiting the relative ranking between the above mixup augmented

samples and their corresponding closer interpolation samples, we specify the unsupervised loss $\mathcal{L}_u(\mathcal{D}_u)$ as follows:

$$\mathcal{L}_u(\mathbf{\Theta}, \mathbf{H}; \mathcal{D}_u) = \frac{1}{N_u^2} \sum_{i=1}^{N_u} \sum_{j=1}^{N_u} \mathbb{I}(\tau_{ij}^u > \tau) \left(\ell_r(\underline{q}_{ij}^u, \underline{t}_{ij}^u) + \ell_r(\overline{q}_{ij}^u, \overline{t}_{ij}^u) \right), \tag{5}$$

where $\tau_{ij}^u = \max(q_{ij}^u, 1 - q_{ij}^u)$, $q_{ij}^u = h_{\mathbf{H}}(\mathbf{z}_i^u, \mathbf{z}_j^u)$, τ is the threshold to select high-confidence pseudo-ranks,

$$\underline{q}_{ij}^u = h_{\mathbf{H}}(\mathbf{z}_i^u, \underline{\mathbf{z}}_{ij}^u), \quad \overline{q}_{ij}^u = h_{\mathbf{H}}(\overline{\mathbf{z}}_{ij}^u, \mathbf{z}_j^u);$$
$$\underline{\mathbf{z}}_{ij}^u = e_{\mathbf{\Theta}}(\underline{\mathbf{x}}_{ij}^u), \quad \overline{\mathbf{z}}_{ij}^u = e_{\mathbf{\Theta}}(\overline{\mathbf{x}}_{ij}^u),$$

and $\underline{t}_{ij}^u = \mathbb{I}(y_i^u > \underline{y}_{ij}^u), \overline{t}_{ij}^u = \mathbb{I}(\overline{y}_{ij}^u > y_j^u)$ are the corresponding pseudo-ranking targets. However, $y_i^u, \underline{y}_{ij}^u, y_j^u, \overline{y}_{ij}^u$ are unknown, and calculating them may also introduce extra noise and cost, especially when the mixup augmented sample is close to the corresponding interpolation sample. Fortunately, according to the monotonicity theory in Theorem 0.1 (given in the blow subsection), the mixup operation can preserve the relative ranking relationship within the interpolation samples with a high probability. In other words, the relative ranking relationship between mixup augmented samples and the corresponding closer interpolation samples is the same as that of interpolation samples. Thus the pseudo-ranking targets $\underline{t}_{ij}^u, \overline{t}_{ij}^u$ can be given by:

$$\underline{t}_{ij}^u = \overline{t}_{ij}^u = t_{ij}^u = \mathbb{I}(q_{ij}^u > 0.5).$$

Monotonicity in latent feature space By applying the monotonicity theory to the manifold mixup, the latent augmented samples are generated as follows:

$$\underline{\mathbf{z}}_{ij} = \lambda \mathbf{z}_i + (1-\lambda)\mathbf{z}_j, \quad \underline{y}_{ij} = \lambda y_i + (1-\lambda) y_j,$$
$$\overline{\mathbf{z}}_{ij} = (1-\lambda)\mathbf{z}_i + \lambda \mathbf{z}_j, \quad \overline{y}_{ij} = (1-\lambda) y_i + \lambda y_j,$$
$$\lambda = \max(\lambda', 1-\lambda'),\ \lambda' \sim \mathrm{Beta}(\alpha, \alpha),\ \alpha \in (0, \infty), \tag{6}$$

and then the corresponding ranking predictions are obtained by $\underline{q}_{ij} = h_{\mathbf{H}}(\mathbf{z}_i, \underline{\mathbf{z}}_{ij})$, $\overline{q}_{ij} = h_{\mathbf{H}}(\overline{\mathbf{z}}_{ij}, \mathbf{z}_j)$. The unsupervised loss $\mathcal{L}_u(\mathbf{\Theta}, \mathbf{H}; \mathcal{D}_u)$ based on the manifold mixup can be calculated by substituting them into Eq.(5).

Full objective of SSR-RCUS Combining the supervised regression loss Eq.(2), and the unsupervised ranking loss Eq.(5), the full objective of SSR-RCUS is formulated below:

$$\mathcal{L}(\mathbf{\Theta}, \mathbf{W}, \mathbf{H}; \mathcal{D}_l, \mathcal{D}_u) = \mathcal{L}_l(\mathbf{\Theta}, \mathbf{W}; \mathcal{D}_l) + \omega_u \mathcal{L}_u(\mathbf{\Theta}, \mathbf{H}; \mathcal{D}_u) \tag{7}$$

Further, we also introduce the ranking loss over labeled data to exploit their relative ranking relationship:

$$\mathcal{L}_r(\mathbf{\Theta}, \mathbf{H}; \mathcal{D}_l) = \frac{1}{N_l^2} \sum_{i=1}^{N_l} \sum_{j=1}^{N_l} \ell_r(\underline{q}_{ij}^l, \underline{t}_{ij}^l) + \ell_r(\overline{q}_{ij}^l, \overline{t}_{ij}^l) \tag{8}$$

where $\underline{t}_{ij}^l = \overline{t}_{ij}^l = t_{ij}^l = \mathbb{I}(y_i > y_j)$. Then, the final objective of SSR-RCUS is given by:

$$\mathcal{L}(\mathbf{\Theta}, \mathbf{W}, \mathbf{H}; \mathcal{D}_l, \mathcal{D}_u) = \mathcal{L}_l(\mathbf{\Theta}, \mathbf{W}; \mathcal{D}_l) + \omega_u \mathcal{L}_u(\mathbf{\Theta}, \mathbf{H}; \mathcal{D}_u) + \omega_r \mathcal{L}_r(\mathbf{\Theta}, \mathbf{H}; \mathcal{D}_l) \tag{9}$$

Implementation In practice, the parameters $\{\mathbf{\Theta}, \mathbf{W}, \mathbf{H}\}$ of the encoder, regressor, and ranking predictor are optimized by using the stochastic optimization with the SGD or AdamW optimizer. Overall, the iterative training procedure of SSR-RCUS is summarized in *Algorithm 1*.

Algorithm 1: SSR-RCUS with manifold mixup

Input: $\mathcal{D}_l$: labeled training dataset $\mathcal{D}_l = \{(\mathbf{x}_i^l, y_i^l)\}_{i=1}^{N_l}$; $\mathcal{D}_u$: unlabeled training dataset $\mathcal{D}_u = \{\mathbf{x}_i^u\}_{i=1}^{N_u}$; $\{\omega_u, \omega_r\}$: coefficient parameters; α: parameter of Beta distribution; τ: confidence threshold;
Output: $\{\mathbf{\Theta}, \mathbf{W}, \mathbf{H}\}$: parameters of encoder, regressor and ranking predictor

Initialize the parameters $\{\mathbf{\Theta}, \mathbf{W}, \mathbf{H}\}$;
for $t = 0$ **to** T **do**
  for $c = 0$ **to** I **do**
    Sample a mini-batch $\{(\mathbf{x}_i^l, y_i^l)\}_{i=1}^{B_l}$ from $\mathcal{D}_l$ and $\{(\mathbf{x}_i^w, \mathbf{x}_i^s)\}_{i=1}^{B_u}$ from $\mathcal{D}_u$ with $\alpha(\cdot)$ and $\mathcal{A}(\cdot)$;
    Calculate $\{\mathbf{z}_i^l\}_{i=1}^{B_l}$, $\{\mathbf{z}_i^w\}_{i=1}^{B_u}$ and $\{\mathbf{z}_i^s\}_{i=1}^{B_u}$ with $e_{\mathbf{\Theta}}(\cdot)$;
    Calculate $\{p_i^l\}_{i=1}^{B_l}$ based on $\{\mathbf{z}_i^l\}_{i=1}^{B_l}$ with $r_{\mathbf{W}(\cdot)}$;
    Calculate $\mathcal{L}_{reg}(\mathbf{\Theta}, \mathbf{W})$ according to Eq.(2) with $\{p_i^l\}_{i=1}^{B_l}$ and $\{y_i^l\}_{i=1}^{B_l}$;
    Calculate $\{\{(\underline{\mathbf{z}}_{ij}^l, \overline{\mathbf{z}}_{ij}^l)\}_{j=1}^{B_l}\}_{i=1}^{B_l}$ based on $\{\mathbf{z}_i^l\}_{i=1}^{B_l}$, and $\{\{(\underline{\mathbf{z}}_{ij}^u, \overline{\mathbf{z}}_{ij}^u)\}_{j=1}^{B_u}\}_{i=1}^{B_u}$ based on $\{\mathbf{z}_i^s\}_{i=1}^{B_u}$ according to Eq.(6);
    Calculate $\{\{(\underline{q}_{ij}^l, \overline{q}_{ij}^l)\}_{j=1}^{B_l}\}_{i=1}^{B_l}$ based on $\{\{(\underline{\mathbf{z}}_{ij}^l, \overline{\mathbf{z}}_{ij}^l, \mathbf{z}_i^l, \mathbf{z}_j^l)\}_{j=1}^{j=B_l}\}_{i=1}^{B_l}$, and $\{\{(\underline{q}_{ij}^u, \overline{q}_{ij}^u)\}_{j=1}^{B_u}\}_{i=1}^{B_u}$ based on $\{\{(\underline{\mathbf{z}}_{ij}^u, \overline{\mathbf{z}}_{ij}^u, \mathbf{z}_i^s, \mathbf{z}_j^s)\}_{j=1}^{B_u}\}_{i=1}^{B_u}$ with $\underline{q}_{ij} = h_{\mathbf{H}}(\mathbf{z}_i, \underline{\mathbf{z}}_{ij})$, $\overline{q}_{ij} = h_{\mathbf{H}}(\overline{\mathbf{z}}_{ij}, \mathbf{z}_j)$;
    Calculate $\{\{q_{ij}^u\}_{j=1}^{B_u}\}_{i=1}^{B_u}$ based on $\{\mathbf{z}_i^w\}_{i=1}^{B_u}$ with $q_{ij} = h_{\mathbf{H}}(\mathbf{z}_i, \mathbf{z}_j)$;
    Calculate $\{\{(\underline{t}_{ij}^u, \overline{t}_{ij}^u)\}_{j=1}^{B_u}\}_{i=1}^{B_u}$ based on $\{\{q_{ij}^u\}_{j=1}^{B_u}\}_{i=1}^{B_u}$ according to $\underline{t}_{ij}^u = \overline{t}_{ij}^u = t_{ij}^u = \mathbb{I}(q_{ij}^u > 0.5)$;
    Calculate $\mathcal{L}_u(\mathbf{\Theta}, \mathbf{H})$, $\mathcal{L}_r(\mathbf{\Theta}, \mathbf{H})$ and $\mathcal{L}(\mathbf{\Theta}, \mathbf{W}, \mathbf{H})$ with Eqs.(5), (8) and (9);
    Update $\{\mathbf{\Theta}, \mathbf{W}, \mathbf{H}\}$ with SGD or AdamW according to $\mathcal{L}(\mathbf{\Theta}, \mathbf{W}, \mathbf{H})$;
  end for
end for

Mixup Preserving Relative Ranking Relationship

Now, we will show that the relative ranking relationship of the interpolation sample pair will be preserved by its samples and the corresponding mixup augmented sample when the response depends on a small fraction of the features in a monotonic way. Specifically, we consider a single index model with measurement error $y = g(\boldsymbol{\phi}^\top \mathbf{x}) + \epsilon$, where $\boldsymbol{\phi} \in \mathbb{R}^d$, g is a monotonic transformation, and ϵ is a sub-Gaussian random variable. The single index model has been

widely used in econometrics, statistics, and deep learning theory (Ge et al. 2019; Lyu and Li 2019; Yang, Balasubramanian, and Liu 2017). We suppose g is monotonic to model the nearly one-by-one correspondence between features and regression targets.

Theorem 0.1. *Let $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j)$ be drawn from the above single index model where $\mathbf{x} \sim \frac{1}{K}\sum_{k=1}^{K}\mathcal{N}_d(\boldsymbol{\mu}_k, \sigma_{\mathbf{x}}^2\mathbf{I}_d)$ with $\boldsymbol{\mu}_k = \frac{k}{\|\boldsymbol{\phi}\|_2}\boldsymbol{\phi}$ for a fixed positive integer K, the sub-Gaussian random variable $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, and $\mathbf{I}_d$ denotes the d-dimensional identity matrix. And $(\widehat{\mathbf{x}}, \widehat{y})$ is their mixup augmented sample. Suppose g is smooth with $|g'| > c$ for a universal constant $c > 0$. It holds that with probability at least $\Phi(\frac{c(1-\lambda)|k_i-k_j|\|\boldsymbol{\phi}\|_2}{\sqrt{2c^2(1-\lambda)^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2+\sigma_\epsilon^2}})$*

$$\widehat{t}_i = t_{ij} = \mathbb{I}(y_i > y_j),$$

and with probability at least $\Phi(\frac{c\lambda|k_i-k_j|\|\boldsymbol{\phi}\|_2}{\sqrt{2c^2\lambda^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2+\sigma_\epsilon^2}})$

$$\widehat{t}_j = t_{ij} = \mathbb{I}(y_i > y_j),$$

where $\widehat{t}_i = \mathbb{I}(y_i > g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}))$, $\widehat{t}_j = \mathbb{I}(g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) > y_j)$, $\mathbb{I}$ denotes the indicator function, and $\Phi(\cdot)$ is the cumulative distribution function of standard normal distribution. When neglecting the measurement error, it holds that with probability at least $\Phi(\frac{|k_i-k_j|}{\sqrt{2}\sigma_{\mathbf{x}}})$

$$\widehat{t}_i = \widehat{t}_j = t_{ij} = \mathbb{I}(y_i > y_j).$$

Proof. Without loss of generality, we suppose g is monotonically increasing. Then, we have $\boldsymbol{\phi}^\top\mathbf{x} \sim \frac{1}{K}\sum_{k=1}^{K}\mathcal{N}_d(\boldsymbol{\mu}_k^\top\boldsymbol{\phi}, \sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2)$,

$$\begin{aligned} y_i - g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) &= g(\boldsymbol{\phi}^\top\mathbf{x}_i) + \epsilon_i - g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) \\ &= g(\boldsymbol{\phi}^\top\mathbf{x}_i) + \epsilon_i - g(\boldsymbol{\phi}^\top(\lambda\mathbf{x}_i + (1-\lambda)\mathbf{x}_j)) \\ &> c(1-\lambda)\boldsymbol{\phi}^\top(\mathbf{x}_i - \mathbf{x}_j) + \epsilon_i \end{aligned}$$

when $y_i > y_j$, and

$$\begin{aligned} y_i - g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) &= g(\boldsymbol{\phi}^\top\mathbf{x}_i) + \epsilon_i - g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) \\ &= g(\boldsymbol{\phi}^\top\mathbf{x}_i) + \epsilon_i - g(\boldsymbol{\phi}^\top(\lambda\mathbf{x}_i + (1-\lambda)\mathbf{x}_j)) \\ &< c(1-\lambda)\boldsymbol{\phi}^\top(\mathbf{x}_i - \mathbf{x}_j) + \epsilon_i \end{aligned}$$

when $y_i < y_j$. According to properties of Gaussian distribution, it holds that

$$\begin{aligned} &c(1-\lambda)\boldsymbol{\phi}^\top(\mathbf{x}_i - \mathbf{x}_j) + \epsilon_i \\ &\sim \mathcal{N}(c(1-\lambda)\boldsymbol{\phi}^\top(\boldsymbol{\mu}_{k_i} - \boldsymbol{\mu}_{k_j}), 2c^2(1-\lambda)^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2 + \sigma_\epsilon^2). \end{aligned}$$

Thus, we have

$$\begin{aligned} &\mathcal{P}(y_i - g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) > 0) \\ &> \mathcal{P}(c(1-\lambda)\boldsymbol{\phi}^\top(\mathbf{x}_i - \mathbf{x}_j) + \epsilon_i > 0) \\ &= 1 - \Phi(-\frac{c(1-\lambda)\boldsymbol{\phi}^\top(\boldsymbol{\mu}_{k_i} - \boldsymbol{\mu}_{k_j})}{\sqrt{2c^2(1-\lambda)^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2 + \sigma_\epsilon^2}}) \\ &= \Phi(\frac{c(1-\lambda)(k_i - k_j)\|\boldsymbol{\phi}\|_2}{\sqrt{2c^2(1-\lambda)^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2 + \sigma_\epsilon^2}}) \end{aligned}$$

when $y_i > y_j$, and

$$\begin{aligned} &\mathcal{P}(y_i - g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) < 0) \\ &> \mathcal{P}(c(1-\lambda)\boldsymbol{\phi}^\top(\mathbf{x}_i - \mathbf{x}_j) + \epsilon_i < 0) \\ &= \Phi(-\frac{c(1-\lambda)\boldsymbol{\phi}^\top(\boldsymbol{\mu}_{k_i} - \boldsymbol{\mu}_{k_j})}{\sqrt{2c^2(1-\lambda)^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2 + \sigma_\epsilon^2}}) \\ &= \Phi(\frac{c(1-\lambda)(k_j - k_i)\|\boldsymbol{\phi}\|_2}{\sqrt{2c^2(1-\lambda)^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2 + \sigma_\epsilon^2}}) \end{aligned}$$

when $y_i < y_j$. When neglecting the measurement error, $\mathcal{P}(y_i - g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) > 0) > \Phi(\frac{k_i-k_j}{\sqrt{2}\sigma_{\mathbf{x}}})$ when $y_i > y_j$, and $\mathcal{P}(y_i - g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) < 0) > \Phi(\frac{k_j-k_i}{\sqrt{2}\sigma_{\mathbf{x}}})$ when $y_i < y_j$. Note that g is monotonically increasing. Thus, it holds $\mathbb{I}(y_i > g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}})) = \mathbb{I}(y_i > y_j)$ with probability $\Phi(\frac{c(1-\lambda)|k_i-k_j|\|\boldsymbol{\phi}\|_2}{\sqrt{2c^2(1-\lambda)^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2+\sigma_\epsilon^2}})$ (or $\Phi(\frac{|k_i-k_j|}{\sqrt{2}\sigma_{\mathbf{x}}})$ when neglecting measurement error).

Similarly, we can have $\mathbb{I}(g(\boldsymbol{\phi}^\top\widehat{\mathbf{x}}) > y_j) = \mathbb{I}(y_i > y_j)$ with probability $\Phi(\frac{c\lambda|k_i-k_j|\|\boldsymbol{\phi}\|_2}{\sqrt{2c^2\lambda^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2+\sigma_\epsilon^2}})$ (or $\Phi(\frac{|k_i-k_j|}{\sqrt{2}\sigma_{\mathbf{x}}})$ when neglecting measurement error). □

Remark In Theorem 0.1, $\frac{c(1-\lambda)|k_i-k_j|\|\boldsymbol{\phi}\|_2}{\sqrt{2c^2(1-\lambda)^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2+\sigma_\epsilon^2}}$, $\frac{c\lambda|k_i-k_j|\|\boldsymbol{\phi}\|_2}{\sqrt{2c^2\lambda^2\sigma_{\mathbf{x}}^2\|\boldsymbol{\phi}\|_2^2+\sigma_\epsilon^2}}$, $\frac{|k_i-k_j|}{\sqrt{2}\sigma_{\mathbf{x}}} \geq 0$. Thus, it holds $\widehat{t}_j = t_{ij} = \mathbb{I}(y_i > y_j)$ and $\widehat{t}_j = t_{ij} = \mathbb{I}(y_i > y_j)$ with a high probability (> 0.5). The correspondence probability will be higher when the difference between y_i and y_j is bigger. In other words, the mixup (including manifold mixup) can preserve the relative ranking relationship of the interpolation sample pair, especially for ones with a large gap between their regression targets. When the transformation is strongly nonlinear and not one-to-one, we can employ the manifold mixup and control the parameter α of the Beta distribution (*e.g.*, $\alpha = 0.5$) to obtain a mixup augmented sample close to one within the sample pair, so as to exploit the local monotonicity of g. Besides, Theorem 0.1 just provides a probabilistic guarantee that the ranking relationship is preserved by mixup with high probability, especially when the difference in targets is meaningful. This isn't an arbitrary constraint but a mathematically grounded foundation.

Related Works

Modern SSL has coalesced around two pillars: consistency regularization and pseudo-labeling. Consistency regularization enhances the model's generalization by applying perturbations to the input and encouraging consistent predictions. Alternatively, pseudo-labeling leverages an initially trained model to generate labels for unlabeled data. By selecting high-confidence predictions as pseudo-labels, this approach effectively transforms unlabeled data into labeled data, thereby expanding the training set and improving model performance. For example, MixMatch (Berthelot et al. 2019) generates soft pseudo-labels by applying multiple augmentations and sharpening the averaged predictions. Then Mixup (Zhang et al. 2018) is used to mix pseudo-labeled samples with labeled

Dataset	Target Range	#Original	#Dev	#Test	Area	Backbone
UTKFace	[1, 116]	18,964	–	4,741	Image Age	not Pre-trained ResNet-28-2
Yelp Review	[0, 4]	250,000	25,000	50,000	Text Opinion	Pre-trained Bert-Small
BVCC	[1, 5]	4,974	1,066	1,066	Audio	Pre-trained Whisper-base

Table 1: Statistics of datasets. *Target Range*: the range of regression targets, *#Original*: the number of original training instances, *#Dev*: the number of development texts, *#Test*: the number of texts for testing, *Area*: the domain of dataset.

samples for training. FixMatch (Sohn et al. 2020) combines pseudo-labeling based on weak augmentation with consistency training on strongly augmented versions. FlexMatch (Zhang et al. 2021a) introduces Curriculum Pseudo Labeling with class-wise increasing thresholds, dynamically adjusting the confidence thresholds based on the model's learning progress for each class.

Semi-supervised regression methods typically build on the smoothness assumption (vanEngelen and Hoos 2020; Yang et al. 2023) that function outputs should vary slowly over high-density regions of the input space and enforce this via consistency regularization. For example, Π-Model (Laine and Aila 2017) enforces consistency by applying two stochastic augmentations to each unlabeled input and minimizing the squared difference between their predictions. Mean Teacher (Tarvainen and Valpola 2017) enhances stability by maintaining an exponential moving average teacher model to guide the student's predictions on perturbed inputs. UCVME (Dai, Li, and Cheng 2023) co-trains Bayesian neural networks with an uncertainty-consistency loss and variational ensembling for robust pseudo-labels. CLSS (Dai et al. 2023) adapts contrastive learning to regression by leveraging recovered ordinal relationships. RankUp (Huang, Fu, and Tsao 2024) transforms regression into a pairwise ranking classification task to leverage semi-supervised classification methods.

The mixup techniques (Zhang et al. 2018; Verma et al. 2019b) produce augmented samples by linear interpolations of training samples based on the linear correspondence assumption. It is simple but can effectively improve robustness and generalization even with scarce and noisy supervision (Thulasidasan et al. 2019; Carratino et al. 2020; Zhang et al. 2021b; Li et al. 2022). Mixup has shown remarkable success in diverse domains, and is currently drawing more attention in continuous domains, *e.g.*, regression tasks, where the model may be strongly nonlinear. The mainstream of methods focuses on how to limit the mixing in regression. RegMix (Hwang and Whang 2022) and C-Mixup (Yao et al. 2022) determine the mixing sequence and probability by employing feature distances and label distances, respectively. RC-Mixup (Hwang, Kim, and Whang 2024) further integrates C-Mixup with multi-round robust training techniques to select clean samples multi-roundly so as to make it robust against noise. Anchor Data Augmentation (Schneider, Goshtasbpour, and Pérez-Cruz 2023) clusters training samples and augments them either towards or away from the cluster centroids with mixup. R-Mixup (Kan et al. 2023) focuses on improving model performance on biological networks. In comparison, our SSR-RCUS preserves the relative ranking relationship between close samples to improve the regression performance based on the proposed theorem, where mixup can keep the relative ranking relationship of the interpolation sample pair, especially for ones with a large gap between their regression targets. It does not limit the mixing, does not depend on a large amount of clean supervision, and avoids the extra sample selection.

Experiments

In this section, we show the experimental settings and the main empirical results.

Experimental Settings

Datasets In the experiments, we employ 3 commonly used datasets from different domains, including *UTKFace* (Zhang, Song, and Qi 2017) (image age estimation), *Yelp Review* (text opinion mining), and *BVCC* (Cooper and Yamagishi 2021) (audio quality assessment)[2]. We utilize the original training/dev/test split of datasets. For all datasets, we construct the labeled dataset $\mathcal{D}_l$ by randomly drawing samples from the corresponding training dataset, treat the remainder as unlabeled dataset $\mathcal{D}_u$, and employ the test dataset for the performance evaluation. The dataset statistics and split information are described in Table 1.

Baseline methods We employ seven semi-supervised regression methods as baselines, including Π-Model (Laine and Aila 2017), Mean Teacher (Tarvainen and Valpola 2017), CLSS (Dai et al. 2023), UCVME (Dai, Li, and Cheng 2023), MixMatch (Berthelot et al. 2019), RankUp (Huang, Fu, and Tsao 2024) and RankUp+RDA (Huang, Fu, and Tsao 2024). Besides, we also use *Supervised* and *Fully Supervised* as two special baselines, which are supervised regression methods, and the former only employs the labeled dataset $\mathcal{D}_l$ for training while the latter utilizes the original training data for training. To compare our SSR-RCUS and baselines fairly, we implement SSR-RCUS and evaluate them based on the RankUp's codebase, which implements all above baselines by modifying the popular semi-supervised classification framework USB (Wang et al. 2022) for regression tasks. For all baselines, we apply the predefined hyperparameters in the codebase. In terms of all comparing methods, we apply the backbones for different datasets as follows: no pre-trained ResNet-28-2 for UTKFace, pre-trained Bert-Small for Yelp Review, and pre-trained Whisper-base for BVCC.

Implementation details For our SSR-RCUS, we set the mixup's parameter $\alpha = 0.5$, and empirically choose the coefficient parameters $\{\omega_u, \omega_r\}$ from $\{0.05, 0.1, 0.2, 0.5, 1.0\}$. The confidence threshold of pseudo-ranks is set to 0.95. We employ the SGD optimizer with learning rate 10^{-2} and

Method	Supervised	Π-Model	Mean Teacher	CLSS	UCVME	MixMatch	RankUp	RankUp+RDA	SSR-RCUS (ours)	Fully Supervised
Number of labeled samples = 10										
R^2 ↑	$-0.141_{\pm 0.102}$	$-0.214_{\pm 0.104}$	$-0.123_{\pm 0.048}$	$-0.195_{\pm 0.152}$	$0.009_{\pm 0.134}$	$0.016_{\pm 0.034}$	$0.334_{\pm 0.061}$	$0.195_{\pm 0.057}$	$\mathbf{0.609}_{\pm 0.053}$	$0.875_{\pm 0.000}^{*}$
MAE↓	$15.74_{\pm 0.67}$	$16.12_{\pm 1.13}$	$15.62_{\pm 0.32}$	$15.59_{\pm 0.98}$	$14.72_{\pm 1.23}$	$14.04_{\pm 0.56}$	$11.45_{\pm 0.76}$	$12.57_{\pm 0.78}$	$\mathbf{8.63}_{\pm 0.57}$	$4.85_{\pm 0.01}^{*}$
SRCC↑	$0.179_{\pm 0.083}$	$0.143_{\pm 0.103}$	$0.165_{\pm 0.033}$	$0.165_{\pm 0.089}$	$0.205_{\pm 0.159}$	$0.412_{\pm 0.047}$	$0.599_{\pm 0.012}$	$0.543_{\pm 0.013}$	$\mathbf{0.797}_{\pm 0.009}$	$0.910_{\pm 0.001}^{*}$
Number of labeled samples = 50										
R^2 ↑	$0.090_{\pm 0.092}^{*}$	$0.100_{\pm 0.086}^{*}$	$0.127_{\pm 0.037}^{*}$	$0.138_{\pm 0.101}^{*}$	$0.157_{\pm 0.110}^{*}$	$0.401_{\pm 0.028}^{*}$	$0.514_{\pm 0.043}^{*}$	$0.552_{\pm 0.041}^{*}$	$\mathbf{0.727}_{\pm 0.042}$	$0.875_{\pm 0.000}^{*}$
MAE↓	$14.13_{\pm 0.56}^{*}$	$13.82_{\pm 1.02}^{*}$	$13.92_{\pm 0.20}^{*}$	$13.61_{\pm 0.92}^{*}$	$13.49_{\pm 0.95}^{*}$	$11.44_{\pm 0.45}^{*}$	$9.96_{\pm 0.62}^{*}$	$9.33_{\pm 0.54}^{*}$	$\mathbf{7.44}_{\pm 0.43}$	$4.85_{\pm 0.01}^{*}$
SRCC↑	$0.371_{\pm 0.071}^{*}$	$0.387_{\pm 0.092}^{*}$	$0.423_{\pm 0.023}^{*}$	$0.447_{\pm 0.074}^{*}$	$0.412_{\pm 0.127}^{*}$	$0.674_{\pm 0.035}^{*}$	$0.832_{\pm 0.008}^{*}$	$0.770_{\pm 0.009}^{*}$	$\mathbf{0.824}_{\pm 0.007}$	$0.910_{\pm 0.001}^{*}$
Number of labeled samples = 250										
R^2 ↑	$0.540_{\pm 0.014}^{*}$	$0.534_{\pm 0.030}^{*}$	$0.586_{\pm 0.020}^{*}$	$0.586_{\pm 0.016}^{*}$	$0.626_{\pm 0.006}^{*}$	$0.692_{\pm 0.013}^{*}$	$0.751_{\pm 0.011}^{*}$	$0.782_{\pm 0.012}^{*}$	$\mathbf{0.791}_{\pm 0.008}$	$0.875_{\pm 0.000}^{*}$
MAE↓	$9.42_{\pm 0.16}^{*}$	$9.45_{\pm 0.30}^{*}$	$8.85_{\pm 0.25}^{*}$	$9.10_{\pm 0.15}^{*}$	$8.63_{\pm 0.17}^{*}$	$7.95_{\pm 0.15}^{*}$	$7.06_{\pm 0.11}^{*}$	$6.57_{\pm 0.18}^{*}$	$\mathbf{6.53}_{\pm 0.15}$	$4.85_{\pm 0.01}^{*}$
SRCC↑	$0.712_{\pm 0.010}^{*}$	$0.706_{\pm 0.015}^{*}$	$0.745_{\pm 0.013}^{*}$	$0.737_{\pm 0.014}^{*}$	$0.767_{\pm 0.007}^{*}$	$0.832_{\pm 0.008}^{*}$	$0.835_{\pm 0.008}^{*}$	$\mathbf{0.856}_{\pm 0.005}^{*}$	$\mathbf{0.856}_{\pm 0.006}$	$0.910_{\pm 0.001}^{*}$
Number of labeled samples = 2000										
R^2 ↑	$0.794_{\pm 0.004}^{*}$	$0.790_{\pm 0.006}^{*}$	$0.793_{\pm 0.004}^{*}$	$0.794_{\pm 0.003}^{*}$	$0.821_{\pm 0.007}^{*}$	$0.824_{\pm 0.004}^{*}$	$0.838_{\pm 0.003}^{*}$	$0.844_{\pm 0.004}^{*}$	$\mathbf{0.846}_{\pm 0.003}$	$0.875_{\pm 0.000}^{*}$
MAE↓	$6.28_{\pm 0.06}^{*}$	$6.31_{\pm 0.10}^{*}$	$6.29_{\pm 0.03}^{*}$	$6.29_{\pm 0.01}^{*}$	$5.90_{\pm 0.07}^{*}$	$6.03_{\pm 0.07}^{*}$	$5.61_{\pm 0.07}^{*}$	$5.51_{\pm 0.07}^{*}$	$\mathbf{5.48}_{\pm 0.05}$	$4.85_{\pm 0.01}^{*}$
SRCC↑	$0.862_{\pm 0.001}^{*}$	$0.860_{\pm 0.003}^{*}$	$0.862_{\pm 0.001}^{*}$	$0.862_{\pm 0.001}^{*}$	$0.877_{\pm 0.002}^{*}$	$0.883_{\pm 0.002}^{*}$	$0.887_{\pm 0.003}^{*}$	$0.890_{\pm 0.003}^{*}$	$\mathbf{0.894}_{\pm 0.002}$	$0.910_{\pm 0.001}^{*}$

Table 2: Regression performance with various numbers of labeled samples on *UTKFace*. The best results are highlighted in boldface.

Method	Supervised	Π-Model	Mean Teacher	CLSS	UCVME	MixMatch	RankUp	RankUp+RDA	SSR-RCUS (ours)	Fully Supervised
Number of labeled samples = 10										
R^2 ↑	$-0.053_{\pm 0.097}$	$-0.055_{\pm 0.102}$	$-0.055_{\pm 0.043}$	$-0.048_{\pm 0.132}$	$-0.025_{\pm 0.101}$	$-0.068_{\pm 0.078}$	$0.077_{\pm 0.059}$	$0.080_{\pm 0.062}$	$\mathbf{0.279}_{\pm 0.054}$	$0.799_{\pm 0.002}^{*}$
MAE↓	$1.240_{\pm 0.097}$	$1.232_{\pm 0.099}$	$1.232_{\pm 0.046}$	$1.189_{\pm 0.102}$	$1.219_{\pm 0.107}$	$1.205_{\pm 0.059}$	$1.127_{\pm 0.073}$	$1.109_{\pm 0.069}$	$\mathbf{0.985}_{\pm 0.065}$	$0.418_{\pm 0.003}^{*}$
SRCC↑	$0.108_{\pm 0.085}$	$0.156_{\pm 0.093}$	$0.156_{\pm 0.054}$	$0.238_{\pm 0.098}$	$0.122_{\pm 0.091}$	$0.226_{\pm 0.047}$	$0.321_{\pm 0.015}$	$0.353_{\pm 0.013}$	$\mathbf{0.545}_{\pm 0.014}$	$0.896_{\pm 0.001}^{*}$
Number of labeled samples = 50										
R^2 ↑	$0.248_{\pm 0.083}$	$0.252_{\pm 0.085}$	$0.235_{\pm 0.036}$	$0.202_{\pm 0.093}$	$0.227_{\pm 0.095}$	$0.204_{\pm 0.037}$	$0.525_{\pm 0.049}$	$0.412_{\pm 0.045}$	$\mathbf{0.548}_{\pm 0.046}$	$0.799_{\pm 0.002}^{*}$
MAE↓	$0.989_{\pm 0.076}$	$1.007_{\pm 0.093}$	$1.016_{\pm 0.043}$	$1.004_{\pm 0.087}$	$1.039_{\pm 0.086}$	$1.018_{\pm 0.036}$	$0.786_{\pm 0.037}$	$0.854_{\pm 0.042}$	$\mathbf{0.760}_{\pm 0.041}$	$0.418_{\pm 0.003}^{*}$
SRCC↑	$0.512_{\pm 0.078}$	$0.509_{\pm 0.081}$	$0.505_{\pm 0.037}$	$0.497_{\pm 0.084}$	$0.500_{\pm 0.079}$	$0.500_{\pm 0.023}$	$0.752_{\pm 0.007}$	$0.667_{\pm 0.009}$	$\mathbf{0.762}_{\pm 0.006}$	$0.896_{\pm 0.001}^{*}$
Number of labeled samples = 250										
R^2 ↑	$0.566_{\pm 0.019}^{*}$	$0.565_{\pm 0.019}^{*}$	$0.565_{\pm 0.019}^{*}$	$0.543_{\pm 0.011}^{*}$	$0.540_{\pm 0.005}^{*}$	$0.381_{\pm 0.008}^{*}$	$0.645_{\pm 0.013}^{*}$	$0.651_{\pm 0.007}^{*}$	$\mathbf{0.714}_{\pm 0.006}$	$0.799_{\pm 0.002}^{*}$
MAE↓	$0.723_{\pm 0.023}^{*}$	$0.730_{\pm 0.024}^{*}$	$0.730_{\pm 0.024}^{*}$	$0.721_{\pm 0.010}^{*}$	$0.775_{\pm 0.006}^{*}$	$0.886_{\pm 0.004}^{*}$	$0.661_{\pm 0.018}^{*}$	$0.632_{\pm 0.009}^{*}$	$\mathbf{0.579}_{\pm 0.008}$	$0.418_{\pm 0.003}^{*}$
SRCC↑	$0.769_{\pm 0.010}^{*}$	$0.769_{\pm 0.009}^{*}$	$0.769_{\pm 0.009}^{*}$	$0.748_{\pm 0.002}^{*}$	$0.763_{\pm 0.005}^{*}$	$0.660_{\pm 0.004}^{*}$	$0.829_{\pm 0.002}^{*}$	$0.810_{\pm 0.005}^{*}$	$\mathbf{0.848}_{\pm 0.003}$	$0.896_{\pm 0.001}^{*}$
Number of labeled samples = 2000										
R^2 ↑	$0.703_{\pm 0.004}$	$0.703_{\pm 0.005}$	$0.704_{\pm 0.003}$	$0.640_{\pm 0.001}$	$0.695_{\pm 0.002}$	$0.522_{\pm 0.001}$	$0.735_{\pm 0.003}$	$0.730_{\pm 0.001}$	$\mathbf{0.747}_{\pm 0.002}$	$0.799_{\pm 0.002}^{*}$
MAE↓	$0.583_{\pm 0.003}$	$0.581_{\pm 0.004}$	$0.581_{\pm 0.002}$	$0.604_{\pm 0.002}$	$0.593_{\pm 0.001}$	$0.774_{\pm 0.003}$	$0.549_{\pm 0.002}$	$0.557_{\pm 0.002}$	$\mathbf{0.529}_{\pm 0.003}$	$0.418_{\pm 0.003}^{*}$
SRCC↑	$0.840_{\pm 0.002}$	$0.840_{\pm 0.003}$	$0.840_{\pm 0.001}$	$0.807_{\pm 0.003}$	$0.836_{\pm 0.002}$	$0.740_{\pm 0.003}$	$0.859_{\pm 0.002}$	$0.855_{\pm 0.001}$	$\mathbf{0.865}_{\pm 0.001}$	$0.896_{\pm 0.001}^{*}$

Table 3: Regression performance with various numbers of labeled samples on *Yelp Review*. The best results are highlighted in boldface.

weight decay 5×10^{-3} for UTKFace, the AdamW optimizer with learning rate 10^{-5} and weight decay 5×10^{-4} for Yelp Review, and the AdamW optimizer with learning rate 2×10^{-6} and weight decay 2×10^{-5} for BVCC. The epoch number is set to 256 for UTKFace, and 100 for Yelp Review and BVCC, with the iteration number 1024 per-epoch. The batch size of labeled data and unlabeled batch ratio are set to 32 and 7 for UTKFace, 8 and 1 for Yelp Review and BVCC. All experiments are carried on a Linux server with 8 NVIDIA 4090 GPUs and 512 GB memory.

Evaluation metrics We employ three commonly used metrics to evaluate the performance of baselines, including R^2, MAE, and SRCC. Given the ground-truth values $\{y_i\}_{i=1}^N$ and the corresponding model's predictions $\{\hat{y}_i\}_{i=1}^N$ of N test samples, the details of metrics are given below:

- **Coefficient of Determination** (R^2) evaluates the proportion of variance in the data explained by the model:

$$R^2 = 1 - \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{\sum_{i=1}^N (y_i - \bar{y})^2}$$

  where $\bar{y}$ is the mean of ground-truth values.
- **Mean Absolute Error (MAE)** indicates the average absolute difference between predicted values and true ones:

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

Metric	Supervised	Π-Model	Mean Teacher	CLSS	UCVME	MixMatch	RankUp	RankUp+RDA	SSR-RCUS (ours)	Fully Supervised
Number of labeled samples = 10										
$R^2\uparrow$	$-0.035_{\pm 0.082}$	$-0.036_{\pm 0.086}$	$-0.039_{\pm 0.083}$	$-0.033_{\pm 0.079}$	$-0.006_{\pm 0.073}$	$-0.031_{\pm 0.092}$	$0.023_{\pm 0.085}$	$-0.120_{\pm 0.089}$	$\mathbf{0.174}_{\pm 0.077}$	$0.764_{\pm 0.002}^{*}$
MAE↓	$0.768_{\pm 0.067}$	$0.772_{\pm 0.069}$	$0.770_{\pm 0.065}$	$0.772_{\pm 0.069}$	$0.762_{\pm 0.057}$	$0.761_{\pm 0.069}$	$0.738_{\pm 0.073}$	$0.787_{\pm 0.065}$	$\mathbf{0.684}_{\pm 0.057}$	$0.351_{\pm 0.003}^{*}$
SRCC↑	$-0.001_{\pm 0.072}$	$-0.018_{\pm 0.075}$	$-0.008_{\pm 0.073}$	$0.187_{\pm 0.069}$	$0.100_{\pm 0.068}$	$0.070_{\pm 0.071}$	$0.272_{\pm 0.064}$	$0.168_{\pm 0.073}$	$\mathbf{0.444}_{\pm 0.061}$	$0.874_{\pm 0.001}^{*}$
Number of labeled samples = 50										
$R^2\uparrow$	$0.208_{\pm 0.057}$	$0.206_{\pm 0.063}$	$0.211_{\pm 0.052}$	$0.392_{\pm 0.065}$	$0.276_{\pm 0.051}$	$0.218_{\pm 0.087}$	$0.446_{\pm 0.063}$	$\mathbf{0.457}_{\pm 0.067}$	$0.450_{\pm 0.068}$	$0.764_{\pm 0.002}^{*}$
MAE↓	$0.670_{\pm 0.034}$	$0.671_{\pm 0.037}$	$0.669_{\pm 0.035}$	$0.577_{\pm 0.041}$	$0.635_{\pm 0.029}$	$0.659_{\pm 0.043}$	$0.549_{\pm 0.039}$	$\mathbf{0.539}_{\pm 0.040}$	$0.541_{\pm 0.035}$	$0.351_{\pm 0.003}^{*}$
SRCC↑	$0.509_{\pm 0.039}$	$0.509_{\pm 0.041}$	$0.516_{\pm 0.038}$	$0.623_{\pm 0.040}$	$0.543_{\pm 0.037}$	$0.499_{\pm 0.068}$	$0.676_{\pm 0.032}$	$\mathbf{0.684}_{\pm 0.034}$	$0.679_{\pm 0.042}$	$0.874_{\pm 0.001}^{*}$
Number of labeled samples = 250										
$R^2\uparrow$	$0.490_{\pm 0.018}^{*}$	$0.489_{\pm 0.021}^{*}$	$0.492_{\pm 0.018}^{*}$	$0.534_{\pm 0.027}^{*}$	$0.553_{\pm 0.011}^{*}$	$0.353_{\pm 0.044}^{*}$	$0.588_{\pm 0.028}^{*}$	$0.598_{\pm 0.027}^{*}$	$\mathbf{0.631}_{\pm 0.025}$	$0.764_{\pm 0.002}^{*}$
MAE↓	$0.533_{\pm 0.006}^{*}$	$0.534_{\pm 0.008}^{*}$	$0.532_{\pm 0.006}^{*}$	$0.499_{\pm 0.010}^{*}$	$0.498_{\pm 0.003}^{*}$	$0.597_{\pm 0.017}^{*}$	$0.470_{\pm 0.012}^{*}$	$0.463_{\pm 0.013}^{*}$	$\mathbf{0.443}_{\pm 0.011}$	$0.351_{\pm 0.003}^{*}$
SRCC↑	$0.741_{\pm 0.009}^{*}$	$0.740_{\pm 0.009}^{*}$	$0.742_{\pm 0.008}^{*}$	$0.748_{\pm 0.008}^{*}$	$0.774_{\pm 0.008}^{*}$	$0.626_{\pm 0.031}^{*}$	$0.776_{\pm 0.010}^{*}$	$0.783_{\pm 0.011}^{*}$	$\mathbf{0.799}_{\pm 0.012}$	$0.874_{\pm 0.001}^{*}$
Number of labeled samples = 2000										
$R^2\uparrow$	$0.756_{\pm 0.002}$	$0.757_{\pm 0.002}$	$0.757_{\pm 0.001}$	$0.753_{\pm 0.003}$	$\mathbf{0.775}_{\pm 0.001}$	$0.686_{\pm 0.005}$	$0.774_{\pm 0.003}$	$0.774_{\pm 0.002}$	$\mathbf{0.775}_{\pm 0.003}$	$0.764_{\pm 0.002}^{*}$
MAE↓	$0.360_{\pm 0.003}$	$0.359_{\pm 0.002}$	$0.359_{\pm 0.002}$	$0.357_{\pm 0.001}$	$0.346_{\pm 0.001}$	$0.412_{\pm 0.007}$	$0.340_{\pm 0.003}$	$\mathbf{0.338}_{\pm 0.001}$	$0.339_{\pm 0.002}$	$0.351_{\pm 0.003}^{*}$
SRCC↑	$0.872_{\pm 0.004}$	$0.872_{\pm 0.003}$	$0.872_{\pm 0.003}$	$0.867_{\pm 0.002}$	$0.880_{\pm 0.003}$	$0.836_{\pm 0.006}$	$0.880_{\pm 0.004}$	$0.881_{\pm 0.001}$	$\mathbf{0.882}_{\pm 0.003}$	$0.874_{\pm 0.001}^{*}$

Table 4: Regression performance with various numbers of labeled samples on *BVCC*. The best results are highlighted in boldface.

- **Spearman's Rank Correlation Coefficient (SRCC)** measures the correlation between the predicted rankings and the ground-truth rankings:

$$\text{SRCC} = 1 - \frac{6\sum_{i=1}^{N} d_i^2}{N(N^2-1)}$$

where d_i is the difference between the ranks of corresponding variables.

For R^2 and SRCC, the larger value is better (denoted by ↑); for MAE, the smaller value implies better performance (denoted by ↓). All metrics are calculated by using the Scikit-Learn tool (Pedregosa et al. 2011) and Scipy.

Main Results and Analysis

To evaluate the effectiveness of SSR-RCUS, we perform experiments on UTKFace, Yelp Review, and BVCC by varying the number of labeled samples over $\{10, 50, 250, 2000\}$. For each dataset and number of labeled samples, we independently run each comparing method 3 times with different seeds and report the average regression performance in Tables 2, 3, and 4, where the results with $*$ are from the public literature (Huang, Fu, and Tsao 2024).

Overall, we can observe that our SSR-RCUS method performs the highest scores among all comparing methods in most (i.e., 33/36) settings, indicating the superior performance of SSR-RCUS. SSR-RCUS consistently outperforms all baseline methods across all datasets when the labeled samples are scarce (*i.e.*, labels=10) where performance differences are more pronounced, and also achieves competitive and even better performance when the number of labeled samples increases. The results are expected because SSR-RCUS preserves the more fine-grained relative ranking relationship with the simple mixup based on the proposed monotonic theorem. There is only a very little performance drop compared RankUp+RDA when labels=50 on BVCC. The possible reason is that the difference among regression targets of BVCC is too small, leading to a relatively low probability for keeping the relative ranking between augmented samples and corresponding interpolation ones according to Theorem 0.1.

Specifically, in the 10-label setting, SSR-RCUS achieves notable gains over the strongest competitor RankUp, with an 82.3% improvement in R^2, a 24.6% reduction in MAE, and up to 33.1% enhancement in SRCC, which implies that the preserving ranking relationships between close unlabeled samples strategy enables more effective utilization of unlabeled samples. Even under moderate label availability, such as 50 or 250 labels, SSR-RCUS continues to surpass all baselines on both UTKFace and Yelp Review. Although performance gaps narrow as label quantity increases, SSR-RCUS remains competitive even in the 2000-label setting. On all three datasets, it nearly matches the Fully-Supervised method in terms of R^2 and SRCC, and on BVCC, it achieves the highest R^2 (0.775) and highest SRCC (0.882) among all methods. This demonstrates that SSR-RCUS does not overfit nor degrade in performance when more labeled data is available.

Conclusion

In this paper, we propose a novel SSR method named SSR-RCUS. Its basic idea is to formulate an auxiliary ranking task, and ingest closed mixup augmented samples with high-confident pseudo-ranks. It depends on an assumption that mixup can preserve relative ranking relationships, where we prove this assumption holds with a high probability. Accordingly, our SSR-RCUS can simultaneously achieve high-confident pseudo-ranks for the ranking task and sample augmentation with mixup. We conduct extensive experiments to examine the performance of SSR-RCUS on three benchmark datasets from various domains. The empirical results indicate that SSR-RCUS can perform better than the existing SSR baseline methods in most cases, and especially, it can achieve very competitive performance with very few labeled samples.

Acknowledgments

We acknowledge support for this project from the National Science and Technology Major Project (No.2021ZD0112500) and the National Natural Science Foundation of China (No.62276113).

References

Berthelot, D.; Carlini, N.; Goodfellow, I. J.; Papernot, N.; Oliver, A.; and Raffel, C. 2019. MixMatch: A Holistic Approach to Semi-supervised Learning. In *Neural Information Processing Systems*, 5050–5060.

Beyer, L.; Zhai, X.; Oliver, A.; and Kolesnikov, A. 2019. S4L: Self-supervised Semi-supervised Learning. In *IEEE/CVF International Conference on Computer Vision*, 1476–1485.

Carratino, L.; Cissé, M.; Jenatton, R.; and Vert, J. 2020. On Mixup Regularization. *arXiv preprint arXiv:2006.06049*.

Chapelle, O.; Schölkopf, B.; and Zien, A., eds. 2006. *Semi-supervised Learning*. The MIT Press.

Cooper, E.; and Yamagishi, J. 2021. How do Voices from Past Speech Synthesis Challenges Compare Today? *arXiv preprint arXiv:2105.02373*.

Dai, W.; Du, Y.; Bai, H.; Cheng, K.; and Li, X. 2023. Semi-Supervised Contrastive Learning for Deep Regression with Ordinal Rankings from Spectral Seriation. In *Neural Information Processing Systems*.

Dai, W.; Li, X.; and Cheng, K. 2023. Semi-Supervised Deep Regression with Uncertainty Consistency and Variational Model Ensembling via Bayesian Neural Networks. In *AAAI Conference on Artificial Intelligence*, 7304–7313.

Dong, S.; Wang, P.; and Abbas, K. 2021. A Survey on Deep Learning and Its Applications. *Computer Science Review*, 40.

Ge, R.; Kuditipudi, R.; Li, Z.; and Wang, X. 2019. Learning Two-layer Neural Networks with Symmetric Inputs. In *International Conference on Learning Representations*.

Grandvalet, Y.; and Bengio, Y. 2004. Semi-supervised Learning by Entropy Minimization. In *Neural Information Processing Systems*, 529–536.

Huang, P.; Fu, S.; and Tsao, Y. 2024. RankUp: Boosting Semi-Supervised Regression with an Auxiliary Ranking Classifier. In *Neural Information Processing Systems*.

Hwang, S.; Kim, M.; and Whang, S. E. 2024. RC-Mixup: A Data Augmentation Strategy against Noisy Data for Regression Tasks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1155–1165.

Hwang, S.-H.; and Whang, S. E. 2022. RegMix: Data Mixing Augmentation for Regression. *arXiv preprint arXiv:2106.03374*.

Kan, X.; Li, Z.; Cui, H.; Yu, Y.; Xu, R.; Yu, S.; Zhang, Z.; Guo, Y.; and Yang, C. 2023. R-Mixup: Riemannian Mixup for Biological Networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1073–1085.

Kou, Z.; Xie, Y.; Wang, H.; Chen, J.; Wang, J.; Xie, M.-K.; Chen, S.; Jia, Y.; Liu, T.; and Geng, X. 2025a. RankMatch: A Novel Approach to Semi-Supervised Label Distribution Learning Leveraging Rank Correlation between Labels. In *Neural Information Processing Systems*.

Kou, Z.; Xuan, H.; Zhu, J.; Wang, H.; Xie, M.-K.; Wang, C.; Wang, J.; Jia, Y.; and Geng, X. 2025b. Tail-Aware Reconstruction of Incomplete Label Distributions with Low-Rank and Sparse Modeling. *IEEE Transactions on Circuits and Systems for Video Technology*, 1–1.

Laine, S.; and Aila, T. 2017. Temporal Ensembling for Semi-Supervised Learning. In *International Conference on Learning Representations*.

Li, C.; Dai, Y.; Feng, L.; Li, X.; Wang, B.; and Ouyang, J. 2024a. Positive and Unlabeled Learning with Controlled Probability Boundary Fence. In *International Conference on Machine Learning*.

Li, C.; Li, X.; Feng, L.; and Ouyang, J. 2022. Who Is Your Right Mixup Partner in Positive and Unlabeled Learning? In *International Conference on Learning Representations*.

Li, C.; Li, X.; and Ouyang, J. 2021. Semi-supervised text classification with balanced deep representation distributions. In *Annual Meeting of the Association for Computational Linguistics*, 5044–5053.

Li, X.; Liang, S.; Li, C.; Wang, P.; and Gu, F. 2024b. Semi-supervised Multi-label Learning with Balanced Binary Angular Margin Loss. In *Neural Information Processing Systems*, 97884–97906.

Lyu, K.; and Li, J. 2019. Gradient Descent Maximizes the Margin of Homogeneous Neural Networks. In *International Conference on Learning Representations*.

Miyato, T.; Maeda, S.; Koyama, M.; and Ishii, S. 2019. Virtual Adversarial Training: A Regularization Method for Supervised and Semi-supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8): 1979–1993.

Pan, S.; Hu, R.; Fung, S.; Long, G.; Jiang, J.; and Zhang, C. 2020. Learning Graph Embedding with Adversarial Training Methods. *IEEE Transactions on Cybernetics*, 50(6): 2475–2487.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; VanderPlas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830.

Schneider, N.; Goshtasbpour, S.; and Pérez-Cruz, F. 2023. Anchor Data Augmentation. In *Neural Information Processing Systems*.

Sohn, K.; Berthelot, D.; Carlini, N.; Zhang, Z.; Zhang, H.; Raffel, C.; Cubuk, E. D.; Kurakin, A.; and Li, C. 2020. FixMatch: Simplifying Semi-supervised Learning with Consistency and Confidence. In *Neural Information Processing Systems*, 596–608.

Tarvainen, A.; and Valpola, H. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Neural Information Processing Systems*, 1195–1204.

Thulasidasan, S.; Chennupati, G.; Bilmes, J. A.; Bhattacharya, T.; and Michalak, S. 2019. On Mixup Training: Improved Calibration and Predictive Uncertainty for Deep Neural Networks. In *Neural Information Processing Systems*, 13888–13899.

vanEngelen, J. E.; and Hoos, H. H. 2020. A Survey on Semi-supervised Learning. *Machine Learning*.

Verma, V.; Kawaguchi, K.; Lamb, A.; Kannala, J.; Solin, A.; Bengio, Y.; and Lopez-Paz, D. 2019a. Interpolation Consistency Training for Semi-supervised Learning. In *Neural Networks*, 90–106.

Verma, V.; Lamb, A.; Beckham, C.; Najafi, A.; Mitliagkas, I.; Lopez-Paz, D.; and Bengio, Y. 2019b. Manifold Mixup: Better Representations by Interpolating Hidden States. In *International Conference on Machine Learning*, 6438–6447.

Wang, Y.; Chen, H.; Fan, Y.; Sun, W.; Tao, R.; Hou, W.; Wang, R.; Yang, L.; Zhou, Z.; Guo, L.; Qi, H.; Wu, Z.; Li, Y.; Nakamura, S.; Ye, W.; Savvides, M.; Raj, B.; Shinozaki, T.; Schiele, B.; Wang, J.; Xie, X.; and Zhang, Y. 2022. USB: A Unified Semi-supervised Learning Benchmark for Classification. In *Neural Information Processing Systems*.

Wu, F.; Jr., A. H. S.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. Q. 2019. Simplifying Graph Convolutional Networks. In *International Conference on Machine Learning*, 6861–6871.

Xie, Q.; Dai, Z.; Hovy, E.; Luong, T.; and Le, Q. 2020a. Unsupervised Data Augmentation for Consistency Training. In *Neural Information Processing Systems*, 6256–6268.

Xie, Q.; Luong, M.; Hovy, E. H.; and Le, Q. V. 2020b. Self-training with Noisy Student Improves ImageNet Classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10687–10698.

Yang, X.; Song, Z.; King, I.; and Xu, Z. 2023. A Survey on DeepSemi-Supervised Learning. *IEEE Transactions on Knowledge and Data Engineering*.

Yang, Z.; Balasubramanian, K.; and Liu, H. 2017. High-dimensional Non-Gaussian Single Index Models via Thresholded Score Function Estimation. In *International Conference on Machine Learning*, 3851–3860.

Yao, H.; Wang, Y.; Zhang, L.; Zou, J. Y.; and Finn, C. 2022. C-Mixup: Improving Generalization in Regression. In *Neural Information Processing Systems*.

Zhang, B.; Wang, Y.; Hou, W.; Wu, H.; Wang, J.; Okumura, M.; and Shinozaki, T. 2021a. FlexMatch: Boosting Semi-supervised Learning with Curriculum Pseudo Labeling. In *Neural Information Processing Systems*, 18408–18419.

Zhang, H.; Cissé, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018. mixup: Beyond Empirical Risk Minimization. In *International Conference on Learning Representations*.

Zhang, L.; Deng, Z.; Kawaguchi, K.; Ghorbani, A.; and Zou, J. 2021b. How Does Mixup Help With Robustness and Generalization? In *International Conference on Learning Representations*.

Zhang, Z.; Song, Y.; and Qi, H. 2017. Age Progression/Regression by Conditional Adversarial Autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition*, 5810–5818.