Partial-Label Regression

Xin Cheng[1], Dengbao Wang[2], Lei Feng[1]*, Minling Zhang[2], Bo An[3]

[1]College of Computer Science, Chongqing University, Chongqing, China
[2]School of Computer Science and Engineering, Southeast University, Nanjing, China
[3]School of Computer Science and Engineering, Nanyang Technological University, Singapore
xincheng@stu.cqu.edu.cn, lfeng@cqu.edu.cn, {wangdb, zhangml}@seu.edu.cn, boan@ntu.edu.sg

Abstract

Partial-label learning is a popular weakly supervised learning setting that allows each training example to be annotated with a set of candidate labels. Previous studies on partial-label learning only focused on the classification setting where candidate labels are all discrete, which cannot handle continuous labels with real values. In this paper, we provide the first attempt to investigate *partial-label regression*, where each training example is annotated with a set of *real-valued* candidate labels. To solve this problem, we first propose a simple baseline method that takes the average loss incurred by candidate labels as the predictive loss. The drawback of this method lies in that the loss incurred by the true label may be overwhelmed by other false labels. To overcome this drawback, we propose an identification method that takes the least loss incurred by candidate labels as the predictive loss. We further improve it by proposing a progressive identification method to differentiate candidate labels using progressively updated weights for incurred losses. We prove that the latter two methods are *model-consistent* and provide convergence analyses. Our proposed methods are theoretically grounded and can be compatible with any models, optimizers, and losses. Experiments validate the effectiveness of our proposed methods.

Introduction

Due to the difficulty of collecting strong supervision information (i.e., fully labeled datasets) in some real-world scenarios, many weakly supervised learning settings were investigated to deal with weak supervision information. Typical weakly supervised learning settings include semi-supervised learning (Chapelle, Scholkopf, and Zien 2006; Sohn et al. 2020), noisy-label learning (Liu and Tao 2015; Malach and Shalev-Shwartz 2017; Patrini et al. 2017), and positive-unlabeled learning (Elkan and Noto 2008; Niu et al. 2016; Kiryo et al. 2017), and multiple-instance learning (Maron and Lozano-Pérez 1997; Andrews, Tsochantaridis, and Hofmann 2002).

In recent years, another weakly supervised learning setting called *partial-label learning* (PLL) (Cour, Sapp, and Taskar 2011) has received much attention from the machine learning and data mining communities. In PLL, each training example is annotated with a set of candidate labels, only one of which is the true label. Due to the massive label ambiguity and noise in data annotation tasks, PLL has been increasingly used in many real-world applications, such as web mining (Luo and Orabona 2010), multimedia content analysis (Zeng et al. 2013), and automatic image annotations (Chen, Patel, and Chellappa 2018).

The major challenge of PLL lies in label ambiguity, as the true label is concealed in the candidate label set and not directly accessible to the learning algorithm. To tackle this problem, many PLL methods have been proposed. These methods achieved satisfactory performance by using appropriate techniques, such as the expectation-maximization algorithm (Jin and Ghahramani 2003; Wang et al. 2022), the maximum margin criterion (Nguyen and Caruana 2008), metric learning (Gong, Yuan, and Bao 2021a; Liu et al. 2018), the manifold regularization (Zhang and Yu 2015; Zhang, Zhou, and Liu 2016; Gong et al. 2018; Wang, Li, and Zhang 2019), and the self-training strategy (Feng and An 2019; Lv et al. 2020; Feng et al. 2020; Wen et al. 2021).

Despite the effectiveness of previous PLL methods, they only focused on the classification setting where candidate labels are all discrete, which cannot handle continuous labels with real values. In many real-world scenarios, regression tasks that learn with real-valued labels can be commonly encountered. However, *how to learn an effective regression model with a set of real-valued candidate labels is an open problem that still remains unexplored*.

In this paper, we provide the first attempt to investigate *partial-label regression* (PLR), where each training example is annotated with a set of real-valued candidate labels. In order to solve the PLR problem, we make the following contributions:

- We propose a simple baseline method that takes the average loss incurred by candidate labels as the predictive loss to be minimized for model training.
- We propose an identification method that takes the least loss incurred by candidate labels as the predictive loss to be minimized for model training.
- We propose a progressive identification method that differentiates candidate labels by associating their incurred losses with progressively updated weights.

*Corresponding author: Lei Feng <lfeng@cqu.edu.cn>.

- We theoretically show that the identification method and the progressive identification method are *model-consistent*, which indicates that the learned model converges to the optimal model.

Preliminaries

In this section, we briefly introduce preliminary knowledge and studies that are related to our PLR problem.

Partial-Label Learning. PLL is a popular weakly supervised classification problem (Gong, Yuan, and Bao 2021b, 2022), where each training example is annotated with a set of discrete candidate labels, only one of which is the true label. Given such training data, PLL aims to construct a multi-class classifier that predicts the label of unseen test data as accurately as possible. The key challenge of PLL is that the false labels in the candidate label set would mislead the model training process. To tackle this problem, many efforts have been made to disambiguate the ambiguous candidate label set. For example, some methods aim at identifying the true label from the candidate label set by using appropriate techniques such as the maximum margin criterion (Nguyen and Caruana 2008; Yu and Zhang 2016) or class activation value (Zhang et al. 2022). Some iterative methods (Feng and An 2019; Lv et al. 2020; Feng et al. 2020) characterize the different contributions of different candidate labels by using confidence scores and iteratively update the confidence score of each candidate label. Although these PLL methods have achieved satisfactory performance, they only focused on the classification setting where candidate labels are all discrete, which cannot handle continuous labels with real values. To solve this problem, our work focuses on learning a regression model with real-valued candidate labels, which would be more challenging than PLL because the label space becomes infinite when candidate labels are real-valued.

Regression. For the ordinary regression problem, let the feature space be $\mathcal{X} \in \mathbb{R}^d$ and the label space be $\mathcal{Y} \in \mathbb{R}$. Let us denote by $(\boldsymbol{x}, y)$ an example including an instance x and a real-valued label y. Each example $(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}$ is assumed to be independently sampled from an unknown data distribution with probability density $p(\boldsymbol{x}, y)$. For the regression task, we aim to learn a regression model $f : \mathcal{X} \mapsto \mathbb{R}$ that minimizes the following expected risk:

$$R(f) = \mathbb{E}_{p(\boldsymbol{x},y)}[\ell(f(\boldsymbol{x}), y)], \tag{1}$$

where $\mathbb{E}_{p(\boldsymbol{x},y)}$ denotes the expectation over the data distribution $p(\boldsymbol{x}, y)$ and $\ell : \mathbb{R}\times\mathbb{R} \mapsto \mathbb{R}_+$ is a conventional loss function (such as mean squared error and mean absolute error) for regression, which measures how well a model estimates a given real-valued label. As $p(\boldsymbol{x}, y)$ is not available and we are given only a number of training examples $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ that are independently drawn from $p(\boldsymbol{x}, y)$, a common strategy is to minimize the empirical risk

$$\widehat{R}(f) = \frac{1}{n}\sum\nolimits_{i=1}^{n} \ell(f(\boldsymbol{x}_i), y_i), \tag{2}$$

which is called *empirical risk minimization* (Vapnik 1999). It can be clearly seen that such a supervised regression method can only deal with fully labeled data where true labels are provided. In our work, we provide the first attempt to investigate *a novel weakly supervised regression problem called partial-label regression*, where each training example is annotated with a set of real-valued candidate labels.

The Proposed Methods

In this section, we present effective methods to train a regression model from data with a set of real-valued candidate labels. We first propose a simple baseline method that averages the contributions of all the candidate labels. However, this intuitive method does not differentiate the true label for model training, and thus may cause the training process to be misled by false labels in the candidate label set. To overcome this drawback, we further propose two theoretically grounded methods, where one directly identifies the true label with the least loss for model training and the other progressively identifies the true label by associating the loss of each candidate label with properly updated weights.

Notations. Suppose the training set for PLR is denoted by $\{(\boldsymbol{x}_i, S_i)\}_{i=1}^n$ where S_i represents the set of real-valued candidate labels assigned to the instance $\boldsymbol{x}_i \in \mathcal{X}$, and each training example $(\boldsymbol{x}_i, S_i)$ is assumed to be sampled from an unknown data distribution with probability density $p(\boldsymbol{x}, S)$. In the PLR setting, the true label $y_i \in \mathcal{Y}$ of the instance $\boldsymbol{x}_i$ is always contained in its candidate label set S_i, i.e., $y_i \in S_i$. In addition, we also assume that the PLR setting satisfies the condition that the *ambiguity degree* (Cour, Sapp, and Taskar 2011) is less than 1, which is defined as

$$\gamma = \sup_{(\boldsymbol{x},y)\sim p(\boldsymbol{x},y),(\boldsymbol{x},S)\sim p(\boldsymbol{x},S),y'\in\mathcal{Y},y'\neq y} p(y' \in S).$$

As shown in the above equation, the ambiguity degree γ is the maximum probability of an incorrect label y' being contained in the candidate label set S (co-occurring with the true label y). If $\gamma = 1$, we cannot differentiate the true label y from the false label y', since they always appear in the same candidate label set.

The Average Method

An intuitive method to solve the PLR problem is to treat each candidate label equally and average the incurred loss of each candidate label:

$$\ell_{\text{avg}}(f(\boldsymbol{x}), S) = \frac{1}{|S|}\sum\nolimits_{y\in S} \ell(f(\boldsymbol{x}), y). \tag{3}$$

In the above equation, $\ell_{\text{avg}}(\boldsymbol{x}, S)$ can be taken as a loss function specially designed for the PLR problem. In Eq. (3), we take into account the influence of each candidate label and regard the averaged loss as the predictive loss on the PLR example $(\boldsymbol{x}, S)$. The average method is intuitive, while the major drawback of this method lies in that the loss incurred by the true label may be overwhelmed by other false labels in the candidate label set.

The Identification Method

We can find that the drawback of the average method comes from that it does not differentiate the true label from the set

Algorithm 1: The Identification Algorithm

Input: Model f, epoch $T_{\max}$, iteration $I_{\max}$, training set $\widetilde{\mathcal{D}} = \{(\boldsymbol{x}_i, S_i)\}_{i=1}^n$.

Initialize model f;
for $t = 1, 2, \ldots, T_{\max}$ **do**
  Shuffle $\widetilde{\mathcal{D}} = \{(\boldsymbol{x}_i, S_i)\}_{i=1}^n$;
  for $j = 1, \ldots, I_{\max}$ **do**
    Fetch mini-batch $\widetilde{\mathcal{D}}_j$ from $\widetilde{\mathcal{D}}$;
    Update model f with $\widehat{R}_{\min}$ in Eq. (5);
  end for
end for

Output: f.

of real-valued candidate labels. Apart from the true label, there are normally multiple false labels in the candidate label set, hence these false labels may dominate the model training process and thus have huge negative impacts on the learned model. To overcome this drawback, we propose an identification method, which regards the candidate label with the least loss as the true label and only considers the least loss of the identified pseudo label as the predictive loss:

$$\ell_{\min}(f(\boldsymbol{x}), S) = \min_{y \in S} \ell(f(\boldsymbol{x}), y). \tag{4}$$

Then, the expected risk and the empirical risk with our proposed identification method (i.e., $\ell_{\min}$) can be represented as follows:

$$\begin{aligned} R_{\min}(f) &= \mathbb{E}_{p(\boldsymbol{x},S)}[\ell_{\min}(f(\boldsymbol{x}), S)], \\ \widehat{R}_{\min}(f) &= \frac{1}{n} \sum\nolimits_{i=1}^{n} [\ell_{\min}(f(\boldsymbol{x}_i), S_i)]. \end{aligned} \tag{5}$$

By directly minimizing the derived empirical risk $\widehat{R}_{\min}(f)$, we can learn an effective regression model from training data with only real-valued candidate labels. The key idea of the identification method lies in that the model has its own ability to identify the true label through the training process. This idea is quite similar to the small-loss selection strategy used in the noisy-label learning problem (Han et al. 2018; Wei et al. 2020). The pseudo code of the identification method is provided in Algorithm 1.

Model Consistency. We demonstrate that the identification method is *model-consistent*. That is, the model learned by the identification method from data with real-valued candidate labels converges to the optimal model learned from fully supervised data. It is noteworthy that the hypothesis space $\mathcal{F}$ is commonly assumed to be powerful enough (Lv et al. 2020), hence the optimal model in the hypothesis space (i.e., $f^\star = \arg\min_{f \in \mathcal{F}} R(f)$) makes the optimal risk equal to 0 (i.e., $R(f^\star) = 0$). We also adopt this assumption throughout this paper.

Theorem 1. *The model $f^\star_{\min} = \arg\min_{f \in \mathcal{F}} R_{\min}(f)$ learned by the identification method is equivalent to the optimal model $f^\star = \arg\min_{f \in \mathcal{F}} R(f)$.*

Theorem 1 shows that the optimal model learned from fully labeled data can be identified by our identification method given only data with real-valued candidate labels.

Algorithm 2: The Progressive Identification Algorithm

Input: model f, epoch $T_{\max}$, iteration $I_{\max}$, training set $\widetilde{\mathcal{D}} = \{(\boldsymbol{x}_i, S_i)\}_{i=1}^n$.

Initialize model f and $w(\boldsymbol{x}, y) = \frac{1}{|S|},\ \forall j \in S$;
for $t = 1, 2, \ldots, T_{\max}$ **do**
  Shuffle $\widetilde{\mathcal{D}} = \{(\boldsymbol{x}_i, S_i)\}_{i=1}^n$;
  for $j = 1, \ldots, I_{\max}$ **do**
    Fetch mini-batch $\widetilde{\mathcal{D}}_j$ from $\widetilde{\mathcal{D}}$;
    Update model f with $\widehat{R}_{\mathrm{wet}}(f)$ in Eq. (7);
  end for
  Update $w(\boldsymbol{x}, y)$ by Eq. (8);
end for

Output: f.

Convergence Analysis. Here, we provide a convergence analysis for the above identification method, which shows that the model $\widehat{f}_{\min} = \arg\min_{f \in \mathcal{F}} \widehat{R}_{\min}(f)$ (empirically learned from only data with real-valued candidate labels by using our identification method) can converge to the optimal model $f^\star$. Given such a convergence analysis, we can observe that the identification method could benefit from the increasing number of training data with real-valued candidate labels. To ensure that $\widehat{f}_{\min}$ converges to $f^\star$, we can show that $R_{\min}(\widehat{f}_{\min})$ converges to $R_{\min}(f^\star)$. Since we have proved the model consistency of the identification method (i.e., $f^\star = f^\star_{\min}$), we can turn to show that $R_{\min}(\widehat{f}_{\min})$ converges to $R_{\min}(f^\star_{\min})$.

Theorem 2. *Suppose the pseudo-dimensions of $\{\boldsymbol{x} \mapsto \ell(f(\boldsymbol{x}), y) \mid f \in \mathcal{F}, y \in \mathcal{Y}\}$ is finite and there exists a constant $M \leq \infty$ such that $|\ell(f(\boldsymbol{x}), y)| \leq M$ for all $(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}$ and $f \in \mathcal{F}$. Then, the estimation error $R_{\min}(\widehat{f}_{\min}) - R_{\min}(f^\star)$ would decrease to zero in the order $\mathcal{O}(1/\sqrt{n})$, where n is the number of data with real-valued candidate labels.*

Theorem 2 demonstrates that the optimal model can be learned by the identification method when the number of training data for PLR approaches infinity.

The Progressive Identification Method

We have introduced the average method and the identification method earlier in this section. The first one treats all the candidate targets equally and the second one focuses too much on a single candidate label, hence both of them fail consider different contributions of candidate labels. To remedy this issue, we further propose a progressive identification method that takes the weighted loss incurred by candidate labels as the predictive loss:

$$\ell_{\mathrm{wet}}(f(\boldsymbol{x}), S) = \sum\nolimits_{y \in S} w(\boldsymbol{x}, y) \ell(f(\boldsymbol{x}), y),$$

where $w(\boldsymbol{x}, y)$ is a weighting function that describes the importance degree of the label y to the instance $\boldsymbol{x}$. Hence in the PLR task, for each training instance $\boldsymbol{x}$, $w(\boldsymbol{x}, y)$ is expected

to satisfy the following conditions:

$$\forall y \in S, w(\boldsymbol{x}, y) \geq 0 \text{ and } \sum\nolimits_{y \in S} w(\boldsymbol{x}, y) = 1. \quad (6)$$

Eq. (6) implies that every candidate label may have an impact on model training, since each of them has the probability of being the true label while only one of them is the true label. It is noteworthy that by manually setting $w(\boldsymbol{x}, y)$ to different values, the progressive identification method can recover the average method and the identification method.

Then, the expected risk and the empirical risk with our proposed progressive identification method (i.e., ℓ_{wet}) can be represented as follows:

$$R_{\text{wet}}(f) = \mathbb{E}_{p(\boldsymbol{x},S)}[\ell_{\text{wet}}(f(\boldsymbol{x}), S)],$$
$$\widehat{R}_{\text{wet}}(f) = \frac{1}{n} \sum\nolimits_{i=1}^{n} [\ell_{\text{wet}}(f(\boldsymbol{x}_i), S_i)], \quad (7)$$

By directly minimizing the derived empirical risk $\widehat{R}_{\text{wet}}(f)$, we can learn an effective model from training data with only real-valued candidate labels. The pseudo code of the identification method is provided in Algorithm 2.

Model Consistency. We show that the progressive identification method is also model-consistent.

Theorem 3. *The model $f^{\star}_{\text{wet}} = \arg\min_{f \in \mathcal{F}} R_{\text{wet}}(f)$ is equivalent to the optimal model $f^{\star} = \arg\min_{f \in \mathcal{F}} R(f)$.*

Theorem 3 shows that the optimal regression model learned from fully labeled data can be identified by the progressive identification method given only data with real-valued candidate labels.

Convergence Analysis. Here, we also provide a convergence analysis for the identification method. We aim to show that the model $\widehat{f}_{\text{wet}} = \arg\min_{f \in \mathcal{F}} \widehat{R}_{\text{wet}}(f)$ (empirically learned from only data with real-valued candidate labels by using our progressive identification method ℓ_{wet}) can converge to the optimal model $f^{\star}$.

Theorem 4. *With the same conditions in Theorem 2, the estimation error $R_{\text{wet}}(\widehat{f}_{\text{wet}}) - R_{\text{wet}}(f^{\star})$ would decrease to zero in the order $\mathcal{O}(1/\sqrt{n})$, where n is the number of data with real-valued candidate labels.*

Theorem 4 shows that the optimal model can also be learned by the progressive identification method when the number of training data for PLR approaches infinity.

Weighting Function Design. Here, we provide the discussion on the specific choice of the weighting function $w(\boldsymbol{x}, y)$. Motivated by the key idea of the identification method, we also consider that the importance degrees of candidate labels can be reflected by the incurred losses. Specifically, if a candidate label has a smaller loss than other candidate labels, then a larger weight (importance degree) should be assigned to this candidate label. Besides, if the loss of a candidate label approaches 0, we consider this candidate label to be the true label, hence the weight assigned to this candidate label would approach 1, and the weights of other labels would approach 0. Based on this perspective, we design the weighting function as follows:

$$w(\boldsymbol{x}, y) = \begin{cases} \frac{\exp(\beta_2 \cdot \ell(f(\boldsymbol{x}), y)^{-\beta_1})}{\sum_{y' \in S} \exp(\beta_2 \cdot \ell(f(\boldsymbol{x}), y')^{-\beta_1})}, & \text{if } y \in S, \\ 0, & \text{otherwise}, \end{cases} \quad (8)$$

where β_1 and β_2 are two hyper-parameters. For the design of $w(\boldsymbol{x}, y)$ in Eq. (8), we first compute a score of the candidate label y by $\beta_2 \cdot \ell(f(\boldsymbol{x}), y)^{-\beta_1}$, which implies that candidate labels with smaller losses would have larger weights. Then, we use the softmax function to normalize the scores of candidate labels and set the weights of non-candidate labels to zero. In this way, the design of $w(\boldsymbol{x}, y)$ in Eq. (8) satisfies all the conditions in Eq. (6). It is noteworthy that such a design is not unique, and there may exist better designs. We leave the exploration of other designs of the weighting function $w(\boldsymbol{x}, y)$ for future work.

Table 1: Characteristics of the used benchmark datasets.

Dataset	# Train	# Test	# Validation	# Features
Abalone	2507	835	835	8
Airfoil	903	300	300	5
Auto-mpg	236	78	78	7
Housing	304	101	101	13
Concrete	618	206	206	8
Power-plant	5742	1913	1913	4
Cpu-act	4916	1638	1638	21

Experiments

Experimental Setup

Datasets. We use seven widely used benchmark regression datasets including Abalone, Airfoil, Auto-mpg, Housing, Concrete, Power-plant, and Cpu-act. All of these datasets can be downloaded from the UCI Machine Learning Repository[1]. For each dataset, we randomly split the original dataset into training, validation, and test sets by the proportions of 60%, 20%, and 20%, respectively. For each instance (feature vector) $\boldsymbol{x}$, the min-max normalization is used for dimensions with a fixed range of values (i.e., $(\boldsymbol{x} - \min(\boldsymbol{x}))/(\max(\boldsymbol{x}) - \min(\boldsymbol{x}))$, and the one-hot encoding is used to encode dimensions of the discrete type. The remaining continuous dimensions are standardized to have mean 0 and standard deviation 1. The characteristics of these datasets are reported in Table 1. We repeat the sampling-and-training process 10 times on these datasets and record the mean squared error with standard deviation.

Base Models. Since our proposed methods do not rely on a specific model, we train two types of base models including the linear model and three-layer multilayer perceptron (MLP) on the above benchmark datasets, to support the flexibility of our method on the choice of base models. The MLP model is a five-layer (d-20-30-10-1) neural network with the ReLU activation function. For both the linear model and the MLP model, we use the Adam optimization method (Kingma and Ba 2015) with the batch size set to 256 and the number of training epochs set to 1000.

[1] https://archive.ics.uci.edu/

Table 2: Test performance (mean squared error with standard deviation) of each method on the seven benchmark datasets training with the MLP model. The best performance is highlighted in bold.

Datasets	$\|\bar{S}\|$	Supervised	AVGL-MSE	AVGL-MAE	AVGL-Huber	AVGV-MSE	AVGV-MAE	AVGV-Huber	IDent	PIDent
Abalone	2	4.66 (0.54)	9.72 (0.77)	4.66 (0.30)	4.68 (0.29)	9.72 (0.78)	7.36 (0.92)	7.52 (1.05)	4.62 (0.33)	**4.55 (0.21)**
	4		14.42 (0.55)	5.04 (0.30)	5.22 (0.31)	14.08 (0.93)	14.08 (0.93)	14.04 (0.61)	4.66 (0.20)	**4.58 (0.22)**
	8		20.63 (2.13)	7.76 (0.82)	7.94 (0.85)	22.68 (3.88)	22.68 (3.88)	21.28 (2.94)	**4.70 (0.26)**	4.71 (0.29)
	16		25.11 (0.65)	13.77 (0.71)	14.03 (0.77)	26.71 (1.37)	26.71 (1.37)	26.61 (2.63)	**4.90 (0.27)**	**4.90 (0.35)**
Airfoil	2	13.77 (3.27)	23.72 (2.71)	16.66 (2.55)	16.17 (2.21)	23.73 (2.78)	23.27 (3.11)	22.99 (2.44)	15.58 (2.65)	**14.99 (3.45)**
	4		30.79 (3.99)	18.47 (3.08)	18.56 (3.04)	30.83 (3.94)	30.67 (2.68)	30.22 (2.92)	16.23 (3.71)	**16.10 (2.97)**
	8		37.19 (4.25)	24.67 (3.33)	24.40 (1.97)	37.16 (4.27)	39.24 (3.44)	38.19 (3.75)	**17.81 (3.49)**	17.86 (3.13)
	16		42.90 (3.86)	31.43 (4.50)	30.81 (3.78)	43.00 (3.76)	45.38 (3.84)	44.19 (4.81)	**23.41 (5.08)**	24.11 (2.98)
Auto-mpg	2	8.77 (1.95)	21.69 (3.50)	21.69 (3.50)	10.03 (2.57)	21.69 (3.50)	17.73 (4.35)	16.74 (4.93)	9.07 (1.74)	**8.64 (1.84)**
	4		31.66 (4.74)	31.66 (4.74)	13.35 (3.49)	31.67 (4.74)	31.41 (4.22)	30.40 (5.24)	9.10 (2.32)	**9.09 (2.43)**
	8		41.57 (3.62)	17.85 (1.97)	18.13 (2.24)	41.57 (3.62)	45.74 (6.16)	43.65 (4.93)	**9.69 (2.18)**	10.10 (2.81)
	16		51.27 (6.95)	32.11 (6.14)	32.23 (3.20)	51.27 (6.95)	55.06 (7.65)	54.42 (6.82)	12.54 (2.27)	**12.14 (2.44)**
Housing	2	14.48 (3.99)	33.82 (6.96)	18.69 (4.64)	17.37 (4.63)	33.82 (6.93)	32.79 (8.45)	34.03 (8.14)	16.18 (3.30)	**15.55 (3.89)**
	4		48.82 (10.97)	26.78 (7.32)	26.82 (8.18)	48.80 (10.96)	51.87 (9.88)	53.67 (8.75)	21.59 (8.78)	**20.53 (6.97)**
	8		60.87 (7.95)	38.18 (6.81)	39.19 (7.03)	60.90 (7.97)	64.02 (8.21)	62.23 (7.86)	29.82 (13.33)	**26.87 (6.83)**
	16		75.30 (9.38)	52.24 (8.96)	53.06 (8.73)	75.29 (9.40)	84.18 (14.38)	77.00 (9.25)	**39.88 (19.69)**	41.09 (15.24)
Concrete	2	36.49 (3.08)	108.08 (9.93)	46.66 (9.09)	48.62 (6.07)	108.23 (9.76)	107.46 (16.31)	106.00 (16.96)	42.14 (7.68)	**40.48 (10.68)**
	4		151.05 (15.19)	75.49 (22.58)	80.38 (17.05)	151.05 (15.17)	172.01 (20.88)	167.53 (21.83)	45.61 (5.10)	**44.87 (7.30)**
	8		195.40 (14.12)	116.18 (25.99)	118.65 (10.85)	195.32 (14.06)	216.01 (26.26)	213.59 (18.63)	72.09 (22.53)	**63.79 (14.13)**
	16		239.26 (17.05)	186.94 (20.89)	183.64 (19.76)	239.64 (17.47)	268.56 (25.83)	259.05 (20.40)	114.35 (37.81)	**112.02 (39.59)**
Power-plant	2	21.00 (1.06)	64.99 (3.45)	23.67 (1.06)	23.43 (1.20)	64.99 (3.45)	42.56 (2.76)	42.68 (2.67)	21.09 (0.96)	**21.07 (1.01)**
	4		104.61 (6.27)	29.24 (1.43)	29.47 (1.17)	104.61 (6.27)	98.36 (5.64)	98.41 (5.79)	21.29 (1.13)	**21.28 (1.19)**
	8		153.07 (8.47)	153.07 (8.47)	48.49 (2.48)	153.07 (8.47)	167.96 (11.37)	166.21 (6.88)	**21.21 (1.06)**	21.36 (0.89)
	16		204.32 (6.74)	204.32 (6.74)	105.45 (4.79)	204.31 (6.74)	225.97 (5.39)	221.17 (8.00)	**21.31 (0.89)**	22.34 (1.99)
Cpu-act	2	6.56 (0.86)	246.38 (15.60)	9.84 (1.04)	9.62 (0.79)	245.68 (15.12)	172.38 (18.19)	169.26 (18.16)	**6.64 (0.54)**	6.91 (0.69)
	4		461.69 (13.66)	33.15 (5.75)	34.58 (7.32)	461.05 (12.63)	525.78 (17.09)	520.64 (14.98)	**7.00 (1.06)**	7.38 (0.41)
	8		730.38 (16.90)	335.04 (20.47)	333.92 (18.43)	730.93 (17.05)	858.55 (11.33)	852.76 (9.78)	**7.48 (1.32)**	7.90 (0.64)
	16		972.73 (29.53)	738.42 (26.71)	735.89 (24.00)	996.21 (46.98)	1107.80 (54.23)	1106.00 (25.97)	9.48 (2.77)	**9.11 (1.93)**

Candidate Label Set Generation. Since this is the first work on PLR, there is no real-world PLR dataset where each instance is assigned a real-valued candidate label set. Hence we need to artificially generate candidate label sets by using the standard datasets in Table 1. We assume that the generation of candidate label sets is instance-independent, which is a widely used data generation assumption in the weakly supervised learning field (Patrini et al. 2017; Ghosh, Kumar, and Sastry 2017; Ishida et al. 2019; Feng et al. 2020). We fix the size of the candidate label set and independently sample the false label multiple times to form the candidate label set. Formally speaking, let us denote by $\widetilde{y}$ a false label in the non-candidate label, then $\widetilde{y}$ is uniformly sampled from the empirically estimated span of label space in the training set (i.e., $\widetilde{y} \sim U(\min_{i\in[n]} y_i, \max_{i\in[n]} y_i)$). We adopt this uniform distribution because a larger candidate label set means more distractors, making the model more difficult to find the true label. For all the datasets, we denote by $|\bar{S}|$ the number of false labels in the candidate label set and set $|\bar{S}|$ to different values (including 2, 4, 8, and 16) for generating the candidate label set.

Compared Methods. In addition to the average method in Eq. (3) that can serve as a baseline method, we also consider a variant of this method. Specifically, for each instance, we take the averaged value of all candidate labels as the true label, and minimize a conventional regression loss function to train the desired model. We name this variant of the average method AVGV and rename the average method AVGL. It is worth noting that the methods mentioned in this

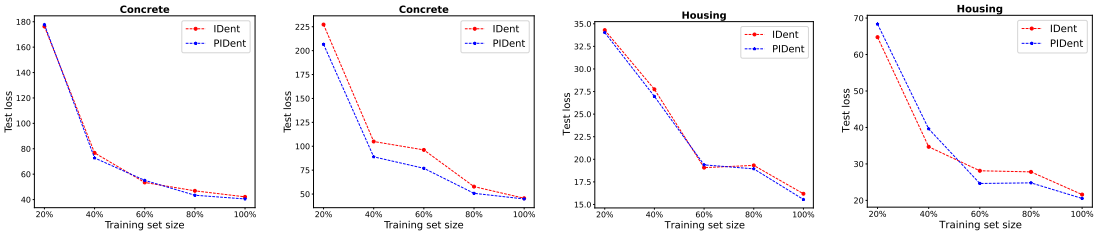
Table 3: Test performance (mean squared error with standard deviation) of each method on the seven benchmark datasets training with the Linear model. The best performance is highlighted in bold.

Datasets	$\lvert\bar{S}\rvert$	Supervised	AVGL-MSE	AVGL-MAE	AVGL-Huber	AVGV-MSE	AVGV-MAE	AVGV-Huber	IDent	PIDent
Abalone	2	5.02 (0.33)	9.78 (0.57)	5.05 (0.30)	5.07 (0.29)	9.78 (0.57)	7.16 (0.55)	7.23 (0.56)	**5.02 (0.32)**	**5.02 (0.30)**
	4		14.57 (0.43)	5.44 (0.28)	5.57 (0.28)	14.57 (0.43)	14.10 (0.99)	14.05 (0.73)	5.10 (0.32)	**5.05 (0.33)**
	8		20.09 (0.84)	7.91 (0.46)	8.03 (0.48)	20.09 (0.84)	21.55 (0.93)	20.43 (0.87)	5.13 (0.36)	**5.09 (0.33)**
	16		25.12 (0.67)	13.78 (0.96)	13.94 (0.70)	25.12 (0.67)	27.21 (0.87)	25.88 (0.68)	5.25 (0.34)	**5.16 (0.35)**
Airfoil	2	23.22 (1.95)	29.22 (2.47)	23.78 (1.98)	23.84 (2.05)	29.22 (2.47)	27.57 (2.30)	27.74 (2.43)	**23.36 (1.91)**	23.38 (1.91)
	4		33.63 (2.86)	25.10 (2.16)	25.12 (2.22)	33.63 (2.86)	33.04 (2.31)	33.28 (2.31)	23.47 (2.02)	**23.43 (2.04)**
	8		39.45 (3.47)	29.13 (2.61)	29.11 (2.47)	39.45 (3.47)	41.17 (3.95)	40.62 (3.81)	24.17 (2.16)	**24.16 (2.05)**
	16		44.36 (3.99)	34.62 (3.41)	34.75 (3.48)	44.36 (3.99)	45.91 (4.41)	45.74 (4.11)	24.74 (2.33)	**24.59 (2.26)**
Auto-mpg	2	10.05 (2.26)	21.44 (3.57)	11.67 (2.93)	11.37 (2.91)	21.38 (3.59)	16.58 (3.97)	16.13 (3.96)	10.16 (2.30)	**10.09 (2.39)**
	4		31.57 (5.60)	13.78 (3.55)	13.64 (3.45)	31.57 (5.00)	31.01 (5.00)	30.48 (5.65)	**11.04 (3.36)**	11.18 (3.33)
	8		41.46 (3.46)	18.45 (2.21)	18.33 (3.95)	41.46 (3.46)	45.65 (6.24)	43.81 (4.90)	11.45 (2.63)	**11.44 (2.35)**
	16		51.06 (6.86)	31.72 (6.24)	32.06 (6.09)	51.06 (6.86)	56.10 (5.94)	54.16 (6.74)	12.73 (3.16)	**12.59 (2.88)**
Housing	2	27.30 (5.99)	37.93 (8.58)	26.30 (5.74)	**26.10 (5.79)**	37.93 (8.58)	36.24 (8.61)	35.28 (8.31)	28.49 (6.88)	28.11 (6.35)
	4		50.52 (11.99)	31.17 (9.42)	31.40 31.40	50.52 (11.99)	53.05 (13.48)	51.51 (12.87)	27.64 (7.15)	**27.38 (6.53)**
	8		60.85 (9.83)	38.85 (8.77)	38.67 (8.91)	60.85 (9.83)	63.91 (8.79)	62.04 (9.23)	32.15 (10.11)	**31.78 (10.37)**
	16		76.78 (11.28)	52.62 (11.59)	52.62 (11.31)	76.78 (11.28)	82.46 (11.58)	80.25 (11.60)	36.56 (9.76)	**35.35 (9.76)**
Concrete	2	110.63 (5.36)	142.29 (8.35)	115.93 (6.92)	115.25 (7.03)	142.29 (8.35)	139.24 (9.83)	138.41 (9.43)	112.38 (5.64)	**112.37 (5.55)**
	4		175.40 (11.53)	126.60 (6.53)	126.32 (6.70)	175.40 (11.53)	182.35 (14.66)	182.02 (14.35)	112.18 (7.22)	**112.11 (7.33)**
	8		209.66 (16.07)	143.17 (8.05)	143.40 (7.91)	209.66 (16.07)	222.84 (16.14)	222.35 (15.76)	115.23 (6.78)	**114.77 (7.39)**
	16		249.88 (15.87)	196.43 (11.66)	196.07 (11.83)	249.88 (15.87)	269.93 (15.50)	268.21 (16.46)	**124.35 (6.75)**	133.28 (27.76)
Power-plant	2	21.41 (0.64)	64.88 (4.91)	23.91 (1.54)	23.95 (1.61)	64.88 (4.91)	43.56 (3.51)	43.85 (3.68)	21.64 (0.73)	**21.62 (0.69)**
	4		104.63 (4.34)	29.70 (1.71)	29.66 (1.73)	104.63 (4.34)	99.26 (7.38)	98.74 (6.36)	21.60 (0.95)	**21.53 (0.95)**
	8		154.75 (7.57)	48.75 (3.32)	48.99 (3.41)	154.75 (7.57)	169.72 (7.12)	166.76 (7.51)	**21.55 (0.70)**	21.85 (0.77)
	16		202.64 (5.41)	106.68 (6.69)	106.91 (7.33)	202.64 (5.41)	225.97 (5.15)	223.18 (5.83)	**22.02 (0.79)**	22.85 (2.09)
Cpu-act	2	98.24 (10.69)	304.66 (22.88)	139.01 19.56	135.64 (17.08)	304.66 (22.88)	238.93 (13.04)	238.35 (12.48)	107.41 (10.86)	**106.11 (9.55)**
	4		514.33 (15.03)	143.62 (13.46)	144.06 (12.78)	514.33 (15.03)	565.37 (13.75)	561.82 (11.68)	116.93 (12.25)	**116.34 (11.46)**
	8		760.87 (57.62)	371.09 (35.76)	359.52 (40.87)	774.19 (21.32)	884.87 (19.49)	858.18 71.00	132.07 (15.29)	**130.64 (15.35)**
	16		1000.14 (18.04)	737.14 (43.63)	762.46 (15.87)	1000.14 (18.04)	1125.31 (19.19)	1131.31 (28.16)	**141.10 (12.60)**	143.12 (15.92)

work can be equipped with arbitrary loss functions. Hence we substitute three regression losses into the AVGL method and the AVGV method, including the *mean squared error* (MSE), the *mean absolute error* (MAE), and the Huber loss. In this way, we can collect six baseline methods, including AVGL-MSE, AVGL-MAE, AVGL-Huber, AVGV-MSE, AVGV-MAE, and AVGV-Huber. We also compare with the supervised regression method that directly trains the model with MSE from fully labeled data (i.e., the true label is provided for each training instance). For the proposed identification method and progressive identification method, we rename them IDent and PIDent, and they are equipped with the commonly used MSE. For AVGL-Huber and AVGV-Huber, the threshold value of the Huber loss is selected from $\{1, 5\}$. For our PIDent method, β_1 is fixed at 0.5 and β_2 is selected from $\{10, 100, 500, 1000, 10000\}$. For all the methods, the learning rate is selected from $\{0.01, 0.001\}$.

Experimental Performance

Table 2 and Table 3 show the mean squared error with standard deviation on the test set using the MLP model and the linear model, respectively. From the two tables, we have the following observations: 1) Our proposed methods IDent and PIDent outperform all the baseline methods, which demonstrates the effectiveness of the two methods. Besides, they could even be on a par with the supervised regression method in some cases, which verifies the ability of the two methods on identifying the true real-valued label. 2) As $|\bar{S}|$ increases, there exists a trend of degraded performance for all the PLR methods. This is because when the number of



(a) $|\bar{S}| = 2$ on Concrete (b) $|\bar{S}| = 4$ on Concrete (c) $|\bar{S}| = 2$ on Housing (d) $|\bar{S}| = 4$ on Housing

Figure 1: The test performance on the Concrete and Housing datasets for the IDent method and the PIDent method when the number of training examples for partial-label regression increases.

false labels in the candidate label set increases, more distracters will be included, and thus the PLR task will become more difficult. It is worth noting that with the increasing of $|\bar{S}|$, the test performance of all the baseline methods degrades dramatically while our proposed methods IDent and PIDent get worse only slightly. 3) By comparing the experimental results reported in Table 2 and 3, we can observe that training with the MLP model is generally better than training with the linear model. This observation is in accordance with the common knowledge that the MLP model is more powerful than the linear model. 4) The PIDent method shows slightly better performance than the IDent method, which implies that the progressive identification strategy (i.e., soft weights) could be more promising than the direct identification strategy (i.e., hard pseudo-labeling). We may expect more significant improvements of the PIDent method over the IDent method with a better designed weighting function. 5) The AVGL methods are generally better than the AVGV methods, which implies that training with average loss is generally better than average value. Besides, MAE and Huber obviously outperform MSE, which shows that the robustness of MAE and Huber still holds in the PLR task.

Performance of Increasing PLR Training Data. As we showed in Theorem 2 and Theorem 4, the models learned by our proposed IDent method and the PIDent method could converge to the optimal model learned from fully labeled data when the number of PLR training examples approaches to infinity. Therefore, the performance of the two methods is expected to be improved if more PLR training examples are provided. To empirically validate such a theoretical finding, we further conduct experiments on the Concrete and Housing datasets by changing the fraction of PLR training examples (100% means that we use all the PLR training examples in the training set). The experimental performance of the IDent method and the PIDent method when we increase the PLR training examples is provided in Figure 1. As shown in Figure 1, the test loss of the two methods generally decreases when more PLR training examples are used for model training. This observation is clearly in accordance with our theoretical analyses in Theorem 2 and Theorem 4, because the learned model would be closer to the optimal model as more PLR training examples are provided.

Conclusion

In this paper, we investigated a novel weakly supervised learning setting called partial-label regression, which is a variant of partial-label learning focusing on the regression task. To solve this problem, we first proposed a simple baseline method that takes the average loss incurred by candidate labels as the predictive loss. To overcome the drawback of this baseline method, we proposed an identification method that takes the least loss incurred by candidate labels as the predictive loss. We further propose a progressive identification method to differentiate candidate labels using progressively updated weights. We proved the model consistency of the latter two methods, which indicates that learned model can converge to the optimal model learned with fully labeled data. Finally, we conducted extensive experiments to demonstrate the effectiveness of our proposed methods. We expect that our first study with simple yet theoretically grounded methods for partial-label regression could inspire more research works on this new task.

We only used a uniform distribution to generate real-valued candidate label sets due to the page limit in the experiments. It would be interesting to further empirically investigate the performance of our proposed methods with other types of generation distributions. Besides, since our proposed methods can be compatible with any models, optimizers, and loss functions, they would be suited for dealing with large-scale regression datasets. Therefore, another promising direction is to apply our proposed methods to solve large-scale problems encountered in real-world application domains such as computer vision (Szeliski 2010) and natural language processing (Chowdhary 2020). In addition, the performance of the PIDent method heavily relies on the designed weighting function. Hence how to properly design an effective weighting function for the PIDent method would be an interesting direction to further improve this method.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (Grant No. 62106028), Chongqing Overseas Chinese Entrepreneurship and Innovation Support Program, and CAAI-Huawei MindSpore Open Fund.

References

Andrews, S.; Tsochantaridis, I.; and Hofmann, T. 2002. Support vector machines for multiple-instance learning. In *NeurIPS*.

Chapelle, O.; Scholkopf, B.; and Zien, A. 2006. *Semi-Supervised Learning*. MIT Press.

Chen, C.-H.; Patel, V. M.; and Chellappa, R. 2018. Learning from ambiguously labeled face images. *TPAMI*, 40(7): 1653–1667.

Chowdhary, K. 2020. Natural language processing. *Fundamentals of Artificial Intelligence*, 603–649.

Cour, T.; Sapp, B.; and Taskar, B. 2011. Learning from partial labels. *Journal of Machine Learning Research*, 12(5): 1501–1536.

Elkan, C.; and Noto, K. 2008. Learning classifiers from only positive and unlabeled data. In *KDD*, 213–220.

Feng, L.; and An, B. 2019. Partial Label Learning with Self-Guided Retraining. In *AAAI*, 3542–3549.

Feng, L.; Lv, J.; Han, B.; Xu, M.; Niu, G.; Geng, X.; An, B.; and Sugiyama, M. 2020. Provably consistent partial-label learning. In *NeurIPS*, 10948–10960.

Ghosh, A.; Kumar, H.; and Sastry, P. 2017. Robust loss functions under label noise for deep neural networks. In *AAAI*.

Gong, C.; Liu, T.; Tang, Y.; Yang, J.; Yang, J.; and Tao, D. 2018. A regularization approach for instance-based superset label learning. *IEEE Transactions on Cybernetics*, 48(3): 967–978.

Gong, X.; Yuan, D.; and Bao, W. 2021a. Discriminative metric learning for partial label learning. *TNNLS*.

Gong, X.; Yuan, D.; and Bao, W. 2021b. Top-k partial label machine. *TNNLS*.

Gong, X.; Yuan, D.; and Bao, W. 2022. Partial Label Learning via Label Influence Function. In *ICML*, 7665–7678.

Han, B.; Yao, Q.; Yu, X.; Niu, G.; Xu, M.; Hu, W.; Tsang, I.; and Sugiyama, M. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, 8527–8537.

Ishida, T.; Niu, G.; Menon, A. K.; and Sugiyama, M. 2019. Complementary-label learning for arbitrary losses and models. In *ICML*, 2971–2980.

Jin, R.; and Ghahramani, Z. 2003. Learning with multiple labels. In *NeurIPS*, 921–928.

Kingma, D. P.; and Ba, J. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Kiryo, R.; Niu, G.; du Plessis, M. C.; and Sugiyama, M. 2017. Positive-unlabeled learning with non-negative risk estimator. In *NeurIPS*, 1674–1684.

Liu, T.; and Tao, D. 2015. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3): 447–461.

Liu, W.; Xu, D.; Tsang, I. W.; and Zhang, W. 2018. Metric learning for multi-output tasks. *TPAMI*, 41(2): 408–422.

Luo, J.; and Orabona, F. 2010. Learning from candidate labeling sets. In *NeurIPS*, 1504–1512.

Lv, J.; Xu, M.; Feng, L.; Niu, G.; Geng, X.; and Sugiyama, M. 2020. Progressive Identification of True Labels for Partial-Label Learning. In *ICML*.

Malach, E.; and Shalev-Shwartz, S. 2017. Decoupling" when to update" from" how to update". In *NeurIPS*, volume 30.

Maron, O.; and Lozano-Pérez, T. 1997. A framework for multiple-instance learning. In *NeurIPS*.

Nguyen, N.; and Caruana, R. 2008. Classification with partial labels. In *KDD*, 551–559.

Niu, G.; du Plessis, M. C.; Sakai, T.; Ma, Y.; and Sugiyama, M. 2016. Theoretical comparisons of positive-unlabeled learning against positive-negative learning. In *NeurIPS*, volume 29.

Patrini, G.; Rozza, A.; Krishna Menon, A.; Nock, R.; and Qu, L. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, 1944–1952.

Sohn, K.; Berthelot, D.; Carlini, N.; Zhang, Z.; Zhang, H.; Raffel, C. A.; Cubuk, E. D.; Kurakin, A.; and Li, C.-L. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 596–608.

Szeliski, R. 2010. *Computer vision: algorithms and applications*. Springer Science & Business Media.

Vapnik, V. N. 1999. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5): 988–999.

Wang, D.-B.; Li, L.; and Zhang, M.-L. 2019. Adaptive Graph Guided Disambiguation for Partial Label Learning. In *KDD*, 83–91.

Wang, H.; Xiao, R.; Li, Y.; Feng, L.; Niu, G.; Chen, G.; and Zhao, J. 2022. PiCO: Contrastive Label Disambiguation for Partial Label Learning. In *ICLR*.

Wei, H.; Feng, L.; Chen, X.; and An, B. 2020. Combating noisy labels by agreement: A joint training method with co-regularization. In *CVPR*, 13726–13735.

Wen, H.; Cui, J.; Hang, H.; Liu, J.; Wang, Y.; and Lin, Z. 2021. Leveraged weighted loss for partial label learning. In *ICML*, 11091–11100.

Yu, F.; and Zhang, M.-L. 2016. Maximum margin partial label learning. In *ACML*, 96–111.

Zeng, Z.-N.; Xiao, S.-J.; Jia, K.; Chan, T.-H.; Gao, S.-H.; Xu, D.; and Ma, Y. 2013. Learning by associating ambiguously labeled images. In *CVPR*, 708–715.

Zhang, F.; Feng, L.; Han, B.; Liu, T.; Niu, G.; Qin, T.; and Sugiyama, M. 2022. Exploiting Class Activation Value for Partial-Label Learning. In *ICLR*.

Zhang, M.-L.; and Yu, F. 2015. Solving the Partial Label Learning Problem: An Instance-Based Approach. In *IJCAI*, 4048–4054.

Zhang, M.-L.; Zhou, B.-B.; and Liu, X.-Y. 2016. Partial label learning via feature-aware disambiguation. In *KDD*, 1335–1344.