Sum of Ranked Range Loss for Supervised Learning

Shu Hu   SHUHU@BUFFALO.EDU
Department of Computer Science and Engineering
University at Buffalo, State University of New York
Buffalo, NY 14260-2500, USA

Yiming Ying   YYING@ALBANY.EDU
Department of Mathematics and Statistics
University at Albany, State University of New York
Albany, NY 12222, USA

Xin Wang   XWANG264@BUFFALO.EDU
Department of Computer Science and Engineering
University at Buffalo, State University of New York
Buffalo, NY 14260-2500, USA

Siwei Lyu*   SIWEILYU@BUFFALO.EDU
Department of Computer Science and Engineering
University at Buffalo, State University of New York
Buffalo, NY 14260-2500, USA

Editor: Lorenzo Rosasco

Abstract

In forming learning objectives, one oftentimes needs to aggregate a set of individual values to a single output. Such cases occur in the aggregate loss, which combines individual losses of a learning model over each training sample, and in the individual loss for multi-label learning, which combines prediction scores over all class labels. In this work, we introduce the s̲um o̲f r̲anked r̲ange (SoRR) as a general approach to form learning objectives. A ranked range is a consecutive sequence of sorted values of a set of real numbers. The minimization of SoRR is solved with the difference of convex algorithm (DCA). We explore two applications in machine learning of the minimization of the SoRR framework, namely the AoRR aggregate loss for binary/multi-class classification at the sample level and the TKML individual loss for multi-label/multi-class classification at the label level. A combination loss of AoRR and TKML is proposed as a new learning objective for improving the robustness of multi-label learning in the face of outliers in sample and labels alike. Our empirical results highlight the effectiveness of the proposed optimization frameworks and demonstrate the applicability of proposed losses using synthetic and real data sets.

Keywords: Learning objective, Aggregate Loss, Rank-based Losses, Multi-class classification, Multi-label classification

1. Introduction

Learning objective is a fundamental component in any machine learning system. In forming learning objectives, we often need to aggregate a set of individual values to a single numerical
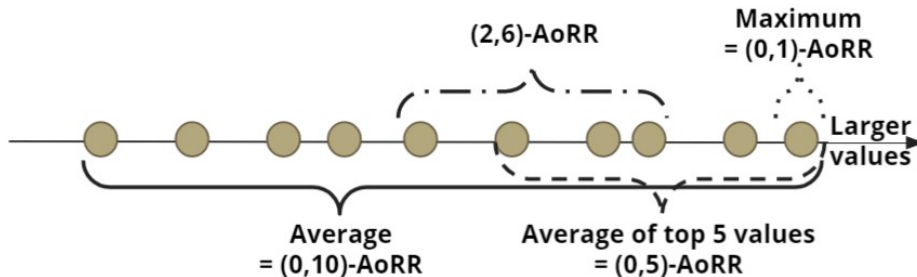
---

*. Corresponding author.

Figure 1: Illustrative examples of different approaches to aggregate individual values to form learning objectives in machine learning. `AoRR` represents the average of ranked range method.

value. Such cases occur in the aggregate loss, which combines individual losses of a learning model over each training sample, and in the individual loss for multi-label learning, which combines prediction scores over all class labels. In this paper, we refer to the loss over all training data as the **aggregate loss**, in order to distinguish it from the **individual loss** that measures the quality of the model on a single training example. For a set of real numbers representing individual values, the ranking order reflects the most basic relation among them. Therefore, designing learning objectives can be achieved by choosing operations defined based on the ranking order of the individual values.

Straightforward choices for such operations are the average and the maximum. Both are widely used in forming aggregate losses (Vapnik, 1992; Shalev-Shwartz and Wexler, 2016) and multi-label losses (Madjarov et al., 2012), yet each has its own drawbacks. The average is *insensitive* to minority sub-groups while the maximum is *sensitive* to outliers, which usually appear as the top individual values. The average top-k loss is introduced as a compromise between the average and the maximum for aggregate loss (Fan et al., 2017) and multi-label individual loss (Fan et al., 2020a). However, it dilutes but not excludes the influences of the outliers. The situation is graphically illustrated in Figure 1.

In this work, we introduce the <u>s</u>um <u>o</u>f <u>r</u>anked <u>r</u>ange (`SoRR`) as a new form of learning objectives that aggregate a set of individual values to a single value. A ranked range is a consecutive sequence of sorted values of a set of real numbers. The `SoRR` can be expressed as the difference between two sums of the top ranked values, which are convex functions themselves. As such, the `SoRR` is the difference of two convex functions, and its optimization is an instance of the difference-of-convex (DC) programming problems (Le Thi and Dinh, 2018). Therefore, it is natural to adopt the existing DC algorithm (DCA) (Tao et al., 1986) to efficiently solve the `SoRR` related learning problems.

We explore two applications in machine learning of the minimization of the `SoRR` framework. The first is to use the *average* of ranked range (`AoRR`) as an aggregate loss for binary/multi-class classification. Specifically, we consider individual logistic loss and individual hinge loss in `AoRR` for binary classification. On the other hand, we consider softmax loss as an individual loss in `AoRR` for multi-class classification. The `AoRR` can be easily obtained by considering using an average operator on `SoRR` (see Section 4 for more details). Unlike previous aggregate losses, the `AoRR` aggregate loss can completely eliminate the influence of outliers if their proportion in training data is known. We also propose a framework that

2

can optimize `AoRR` without prior knowledge about the proportion of outliers by utilizing a clean validation set extracted from the training data set. Second, we use a special case of `SoRR` as a new type of individual loss for multi-label, the *top-k multi-label* (`TKML`) loss, which explicitly encourages the true labels in the top k range. Furthermore, we propose a `TKML-AoRR` loss, which combines the `AoRR` and the `TKML` methods. The proposed loss can eliminate the outliers in the top-k multi-label learning. Then a heuristic algorithm is designed to optimize `TKML-AoRR` . The new learning objectives are tested and compared experimentally on several synthetic and real data sets[1].

The main contributions of this work can be summarized as follows:

- We introduce `SoRR` as a general learning objective and show that it can be formulated as the difference of two convex functions.

- Based on `SoRR`, we introduce the `AoRR` aggregate loss for binary/multi-class classification at the sample level and the `TKML` individual loss for multi-label/multi-class learning at the label level. The two `SoRR`-based losses are further combined to form the `TKML-AoRR` loss that can improve the robustness of multi-label learning in the face of outliers in data and labels alike.

- We also explore several theoretical aspects of `SoRR`-based losses, including connecting `AoRR` to the Condition Value at Risk (CVaR) and explore its generalization property, establishing its classification calibration with regards to the optimal Bayes classifier.

- The learning of `SoRR`-based losses can be generally solved by the DC algorithm, but we also show their connections with the bilevel optimization. Furthermore, we propose practical methods to better determine the hyper-parameters in `SoRR`-based losses.

- We empirically demonstrate the robustness and effectiveness of the proposed `AoRR`, `TKML`, `TKML-AoRR`, and their optimization frameworks on both synthetic and real data sets.

This paper significantly extends our previous conference paper (Hu et al., 2020) both theoretically and experimentally in the following aspects: 1) We establish the relations between `SoRR` and the bilevel optimization problem (Borsos et al., 2020) (Section 3.1); 2) The `AoRR` aggregate loss is reformulated by the Condition Value at Risks (CVaRs) and its generalization property is studied (Section 4.2); 3) We propose a new framework for automatically determining the hyper-parameters in `SoRR`-based losses and empirically demonstrate its effectiveness (Section 4.3 and 4.4.2); 4) We combine `AoRR` aggregate loss at the sample level and `TKML` individual loss at the label level to reduce the influence of the outliers in the top-k multi-label learning procedure. We also propose a heuristic algorithm to optimize this combined loss and verify its effectiveness through experiments (Section 6).

The rest of the paper is organized as follows. In Section 2, we summarize the existing works that are related to this work and discuss how this work differs from them. In Section 3, we define the sum of ranked range `SoRR`, formulate it as a DC problem and provide a DC algorithm for solving it. Then we connect the `SoRR` to the bilevel optimization problem. In Section 4, we propose a new form of aggregate loss named `AoRR` aggregate loss based on

---

1. Code available at `https://github.com/discovershu/SoRR`.

`SoRR` for binary and multi-class classification. We provide its interpretation and study its classification calibration property. We also connect it to the CVaR problem and provide a generalization bound. Furthermore, to make it more reliable in the practice, we propose a new framework for determining the hyper-parameters of `AoRR`. In Section 5, we define `TKML` individual loss based on `SoRR` for multi-label/multi-class learning. In Section 6, we define `TKML-AoRR` loss, which combines `TKML` loss for label level and `AoRR` loss for sample level. The `TKML-AoRR` loss can eliminate the influence of the outliers in the top-k multi-label learning. Then we propose a heuristic algorithm to optimize it. Section 7 concludes the paper with discussions.

2. Related Work

There are a large body of works that focus on designing different forms of individual losses to solve specific problems and studying their properties, especially for binary classification. The earliest work can be traced back to 1960s by Shuford et al. (1966) and Savage (1971). Specifically, Shuford et al. (1966) propose admissible probability measurement procedures as individual losses to measure students' degree-of-belief probabilities in an educational environment. Savage (1971) characterizes scoring rules for probabilistic forecasts of categorical and binary variables. More recent work of Buja et al. (2005); Masnadi-Shirazi and Vasconcelos (2008); Lin et al. (2017); Reid and Williamson (2010) continue to study this topic. For example, Buja et al. (2005) study loss functions for binary class probability estimation and classification. Masnadi-Shirazi and Vasconcelos (2008) discuss the robustness of outliers from a theoretical aspect in designing loss functions for classification. Reid and Williamson (2010) study general composite binary losses. Many researchers prefer to select and use individual losses in terms of a convex function because of the good global convergence properties in its optimization. However, non-convex individual losses are also explored in recent work, such as Lin et al. (2017); He et al. (2010); Wu and Liu (2007); Yu et al. (2010). Concretely, to handle the imbalanced data problem, Lin et al. (2017) design focal loss for object detection. To handle the outliers problem, He et al. (2010) propose to learn a robust sparse representation for face recognition based on the correntropy (Liu et al., 2007) along with the use of an ℓ_1 norm penalty, which is insensitive to outliers. Wu and Liu (2007) propose a robust truncated hinge loss for SVM. Yu et al. (2010) propose a robust estimation method for classification based on loss clipping. All of them are trying to design non-convex individual losses for robust learning.

Rank-based aggregate losses are oftentimes overlooked in existing machine learning literature. One relevant topic is the data subset selection problem (Wei et al., 2015), which is about selecting a subset from a large training dataset to train a model while incurring a minimal average loss, the learning objective of which can be regarded as a special aggregate loss that averages over the individual losses corresponding to data selected into the subset. Hard example mining is an effective method in the training and is widely used in existing works (Gidaris and Komodakis, 2015; Liu et al., 2016; Shrivastava et al., 2016; Lin et al., 2017). For instance, in online hard example mining, Shrivastava et al. (2016) use top-k samples with the largest losses for each training image to update the model parameter. In object detection, Lin et al. (2017) propose a weighted cross-entropy loss, which assigns more weights on candidate bounding box with large losses. They obtain a better performance

with this method than using the conventional cross-entropy loss. However, it should be mentioned that the connections of these works to the aggregate loss are incidental. The most relevant works on the rank-based aggregate loss are Fan et al. (2017) and Lyu et al. (2020). In their works, Fan et al. and Lyu et al. propose an average top-k loss as a type of aggregate loss. They also mention that the maximum loss (Shalev-Shwartz and Wexler, 2016) and the average loss (Vapnik, 1992) are two special cases of their average top-k loss. They demonstrate that their loss can alleviate the influence of data with imbalance and outlier problems in the learning process. However, as we mentioned before, their method can dilute but not completely eliminate the influence of the outliers.

Rank-based individual losses have been studied in many tasks. For example, multi-label/multi-class learning, ranking, and information retrieval. The work from Rudin (2009); Usunier et al. (2009) propose a form of individual loss that gives more weights to the samples at the top of a ranked list. This idea is further extended to multi-label learning in the work of Weston et al. (2011). Top-1 loss is commonly used in multi-class learning, which causes more penalties when the class corresponding to the top-1 prediction score inconsistent with the ground-truth class label (Crammer and Singer, 2001). Since a sample may contain multiple classes and some classes may overlap, Lapin et al. (2015, 2016, 2017) propose top-k multi-class loss that can introduce penalties when the ground-truth label does not appear in the set of top-k labels as measured by their prediction scores. In multi-label learning, many works focus on the ranking-based approaches, such as Zhang and Zhou (2006); Fan et al. (2020b). The loss functions from their methods encourage the predicted relevancy scores of the ground-truth positive labels to be higher than that of the negative ones. Similar to the motivation of the top-k multi-class classification, in multi-label learning, the classifier is expected to include as many true labels as possible in the top k outputs. However, there are no dedicated methods for the top-k multi-label learning. Therefore, in this paper, we propose `TKML` individual loss for multi-label learning to fill this gap.

Rank-based losses are also popular to be used at the sample and label levels simultaneously. For example, the work of Rawat et al. (2020) proposed a doubly-stochastic mining method, which combines the methods of Kawaguchi and Lu (2020) and Lapin et al. (2015). They use the average top-k methods both at the data sample and label levels to construct the final loss function. Then the constructed loss is applied to solve modern retrieval problems, which are characterized by training sets with a huge number of labels, and heterogeneous data distributions across sub-populations. However, their method does not consider the outliers or noisy labels that may exist in real-world data sets. Such problems can make the performance of the proposed model plummet. In addition, their model only works on multi-class problems and cannot be applied to multi-label learning directly. In this paper, we fill this gap and propose a `TKML-AoRR` loss for solving the noise problem in top-k multi-label learning. Furthermore, we design a heuristic algorithm for optimizing `TKML-AoRR` loss and show it can be generalized to other existing algorithms in Shen and Sanghavi (2019); Shah et al. (2020); Kawaguchi and Lu (2020); Rawat et al. (2020).

3. Sum of Ranked Range (SoRR)

For a set of real numbers $S = \{s_1, \cdots, s_n\}$, we use $s_{[k]}$ to denote the *top-k value*, which is the k-th largest value after sorting the elements in S (ties can be broken in any consistent

Symbol	Description
$S = \{s_1, \cdots, s_n\}$	A set of real numbers
$s_{[k]}$ $(s_{[m]})$	The k-th (m-th) largest value after sorting the elements in S
$s_i(\cdot)$	The i-th individual loss
$\phi_k(\cdot)$ $(\phi_m(\cdot))$	The sum of top-k (top-m) values
$\psi_{m,k}(\cdot)$	The sum of (m, k)-ranked range values
q_i	The weight of $s_i(\cdot)$
x	A multi-dimensional data instance
y	A label (class) of x
$f_\theta(\cdot)$, $f(\cdot; \theta)$	A parametric function (classifier) with parameters θ
$\mathcal{L}(\cdot)$	Loss function
$\mathcal{D}$	Training data set
$\widetilde{\mathcal{D}}$	Validation data set
$\mathcal{Y} = \{1, \cdots, l\}$	A set of labels (classes) with size l
Y	Ground truth labels of x and it is a subset of $\mathcal{Y}$

Table 1: Frequently used symbols in this paper.

way). Correspondingly, we define $\phi_k(S) = \sum_{i=1}^{k} s_{[i]}$ as the *sum of the top-k* values of S. For two integers k and m, $0 \le m < k \le n$, the (m, k)-ranked range is the set of sorted values $\{s_{[m+1]}, \cdots, s_{[k]}\}$. The sum of (m, k)-ranked range $((m, k)$-SoRR) is defined as $\psi_{m,k}(S) = \sum_{i=m+1}^{k} s_{[i]}$, and the average of (m, k)-ranked range $((m, k)$-AoRR) is $\frac{1}{k-m}\psi_{m,k}(S)$. It is easy to see that the sum of ranked range (SoRR) is the difference between two sums of top values as, $\psi_{m,k}(S) = \phi_k(S) - \phi_m(S)$. Also, the top-$k$ value corresponds to the $(k-1, k)$-SoRR, as $\psi_{k-1,k}(S) = s_{[k]}$. Similarly, the median can also be obtained from AoRR, as $\frac{1}{\lceil \frac{n+1}{2} \rceil - \lfloor \frac{n+1}{2} \rfloor + 1}\psi_{\lfloor \frac{n+1}{2} \rfloor - 1, \lceil \frac{n+1}{2} \rceil}(S)$. We collect symbols and notations to be used throughout the paper in Table 1.

In machine learning problems, we are interested in the set $S(\theta) = \{s_1(\theta), \cdots, s_n(\theta)\}$ formed from a family of functions where each s_i is a convex function of parameter θ. For example, in practice, s can be a convex loss function. n is the total number of training samples. s_i is the individual loss function of sample i. We can use SoRR to form learning objectives. In particular, we can eliminate the ranking operation and use the equivalent form of SoRR in the following result. Denote $[a]_+ = \max\{0, a\}$ as the hinge function.

Theorem 1 *Suppose $s_i(\theta)$ is convex with respect to θ for any $i \in [1, n]$, then*

$$\min_\theta \psi_{m,k}(S(\theta)) = \min_\theta \left[\min_{\lambda \in \mathbb{R}} \left\{ k\lambda + \sum_{i=1}^{n} [s_i(\theta) - \lambda]_+ \right\} - \min_{\hat{\lambda} \in \mathbb{R}} \left\{ m\hat{\lambda} + \sum_{i=1}^{n} [s_i(\theta) - \hat{\lambda}]_+ \right\} \right]. \quad (1)$$

Furthermore, $\hat{\lambda} > \lambda$, when the optimal solution is achieved.

The proof of Theorem 1 is in Appendix A.1. Note that $\psi_{m,k}(S(\theta))$ is not a convex function of θ. But its equivalence to the difference between $\phi_k(S(\theta))$ and $\phi_m(S(\theta))$ suggests that $\psi_{m,k}(S(\theta))$ is a difference-of-convex (DC) function, because $\phi_k(S(\theta))$ and $\phi_m(S(\theta))$ are convex functions of θ in this setting. As such, a natural choice for its optimization is the DC algorithm (DCA) (Tao et al., 1986).

To be specific, for a general DC problem formed from two convex functions $g(\theta), h(\theta)$ as $\bar{s}(\theta) = g(\theta) - h(\theta)$, DCA iteratively search for a critical point of $\bar{s}(\theta)$ (Thi et al., 2017). At

---

Algorithm 1: DCA for Minimizing SoRR

---
1 **Initialization:** $\theta^{(0)}$, $\lambda^{(0)}$, η_l, and two hyperparameters k and m
2 **for** $t = 0, 1, ...$ **do**
3   Compute $\hat{\theta}^{(t)}$ with equation (3)
4   **for** $l = 0, 1, ...$ **do**
5   Compute $\theta^{(l+1)}$ and $\lambda^{(l+1)}$ with equation (2)
6   **end**
7   Update $\theta^{(t+1)} \leftarrow \theta^{(l+1)}$
8 **end**

---

each iteration of DCA, we form an affine majorization of function h using its sub-gradient at $\theta^{(t)}$, *i.e.*, $\hat{\theta}^{(t)} \in \partial h(\theta^{(t)})$, and then update $\theta^{(t+1)} \in \text{argmin}_\theta \left\{ g(\theta) - \theta^\top \hat{\theta}^{(t)} \right\}$. DCA is a descent method without line search, which means the objective function is monotonically decreased at each iteration (Tao and An, 1997). It does not require the differentiability of $g(\theta)$ and $h(\theta)$ to assure its convergence. Moreover, it is known that DCA converges from an arbitrary initial point and often converges to a global solution (Le Thi and Dinh, 2018). For example, the authors in Tao and An (1997) proved that θ^* is a critical point of $\overline{s}(\theta)$ if $\partial g(\theta^*) \cap \partial h(\theta^*) \neq \emptyset$, or equivalently, $0 \in \partial g(\theta^*) - \partial h(\theta^*)$. A critical point can lead to a local minimizer of $\overline{s}(\theta)$ if it admits a neighborhood U such that $\partial h(\theta) \cap \partial g(\theta^*) \neq \emptyset$, $\forall \theta \in U \cap \text{dom } g$. While a DC problem can be solved based on standard (sub-)gradient descent methods, DCA seems to be more amenable to our task because of its appealing properties and the natural DC structure of our objective function. In addition, as shown in Piot et al. (2016) with extensive experiments, DCA empirically outperforms the gradient descent method on various problems.

To use DCA to optimize SoRR, we need to solve the convex sub-optimization problem

$$\min_\theta \left[\min_\lambda \left\{ k\lambda + \sum_{i=1}^{n} [s_i(\theta) - \lambda]_+ \right\} - \theta^T \hat{\theta} \right].$$

This problem can be solved using a stochastic sub-gradient method (Bottou and Bousquet, 2008; Rakhlin et al., 2011; Srebro and Tewari, 2010). We first randomly sample $s_{i_l}(\theta^{(l)})$ from the collection of $\{s_i(\theta^{(l)})\}_{i=1}^n$ and then perform the following steps:

$$\theta^{(l+1)} \leftarrow \theta^{(l)} - \eta_l \left(\partial s_{i_l}(\theta^{(l)}) \cdot \mathbb{I}_{[s_{i_l}(\theta^{(l)}) > \lambda^{(l)}]} - \hat{\theta}^{(t)} \right), \quad \lambda^{(l+1)} \leftarrow \lambda^{(l)} - \eta_l \left(k - \mathbb{I}_{[s_{i_l}(\theta^{(l)}) > \lambda^{(l)}]} \right), \quad (2)$$

where η_l is the step size. $\mathbb{I}_{[a]}$ is an indicator function with $\mathbb{I}_{[a]} = 1$ if a is true and 0 otherwise. In equation (2), we use the fact that the sub-gradient of $\phi_m(S(\theta))$ is computed, as

$$\hat{\theta} \in \partial \phi_m(S(\theta)) = \sum_{i=1}^{n} \partial s_i(\theta) \cdot \mathbb{I}_{[s_i(\theta) > s_{[m]}(\theta)]}, \quad (3)$$

where $\partial s_i(\theta)$ is the gradient or a sub-gradient of convex function $s_i(\theta)$ (Proof can be found in Appendix A.2)[2]. The pseudo-code of minimizing SoRR is described in Algorithm 1. For a

---

2. For large data sets, we can use a stochastic version of DCA, which is more efficient with a provable convergence to a critical point (Thi et al., 2019).

given outer loop size $|t|$, an inner loop size $|l|$, and training sample size n, the time complexity of Algorithm 1 is $O(|t|(n \log n + |l|))$.

To better understand SoRR, we provide more discussion about it. Since SoRR is non-convex, an intuitive strategy to optimize a non-convex loss function is to use a surrogate convex loss to replace it and apply existing convex optimization approaches to solve the proposed surrogate convex loss. However, there are two drawbacks to this strategy. First, a significant learning property of the surrogate loss is its consistency, which requires the optimal minimizers of the surrogate loss to be near to or exact the optimal minimizers of the original loss. As we shall see, the study of consistency is significantly more complex for SoRR. Fortunately, we rewrite SoRR to a DC problem (equation (1)), and then we can show AoRR aggregate loss (a variant of SoRR only add an additional factor) satisfies the classification calibration (a sufficient condition for consistency) under several moderate conditions in Section 4.1. Second, it is very hard to find a suitable convex loss to surrogate SoRR. If we can find a surrogate convex loss, the optimal solution obtained by learning this convex loss and the actual optimal solution based on SoRR are not guaranteed to be equal. This difference comes not only from the optimization algorithm, but also from the learning objective. For example, we may select the sum of the top-k loss $\phi_k(S)$ to surrogate $\psi_{m,k}(S)$ since it is a convex loss. But $\phi_k(S)$ will use extremely largest values (top-1 value to top-m value) to train the model, while $\psi_{m,k}(S)$ is not. Therefore, they have different meanings in model learning. In other words, the learned models are exactly different. The model learned by using $\phi_k(S)$ may not satisfy the original requirements in specific tasks such as robust to outliers. However, in equation (1), the DC term (right-hand side) is exactly equivalent to the original SoRR formula (left-hand side). Since the DC problem is well studied in the existing works and can be solved by DCA with provable convergence properties, it is beneficial and natural to use the DC function to replace the original ranking formula even if the DC function is also non-convex.

3.1 Connection with Bilevel Optimization

We give another intuitive interpretation of SoRR. Suppose we have 10 individual losses in the ranked list. To extract the sum of $(2, 6)$-ranked range, we can select a subset, which contains the bottom 8 individual losses from the ranked list in the beginning. Then we select the top 4 individual losses from this subset as the finalized $(2, 6)$-ranked range. Suppose $s_i(\theta) \geq 0$, based on this new intuition, we can reformulate SoRR. First, we sum the bottom $n - m$ losses as follows,

$$\sum_{i=m+1}^{n} s_{[i]}(\theta) = \min_{q} \sum_{i=1}^{n} q_i s_i(\theta) \quad \text{s.t. } q_i \in \{0, 1\}, \; ||q||_0 = n - m,$$

where $q = \{q_1, \cdots, q_n\} \in \{0, 1\}^n$, and q_i is an indicator. When $q_i = 0$, it indicates that the i-th individual loss is not included in the objective function. Otherwise, the objective function should include this individual loss. Next, we sum the top-$(k - m)$ individual losses

from the bottom $n - m$ individual losses as follows,

$$\min_{q} \sum_{i=1}^{k-m} (qs(\theta))_{[i]} \quad \text{s.t. } q_i \in \{0, 1\}, \ ||q||_0 = n - m$$

$$= \min_{\lambda, q}(k - m)\lambda + \sum_{i=1}^{n} [q_i s_i(\theta) - \lambda]_+ \quad \text{s.t. } q_i \in \{0, 1\}, \ ||q||_0 = n - m \quad (4)$$

$$= \min_{\lambda, q}(k - m)\lambda + \sum_{i=1}^{n} q_i [s_i(\theta) - \lambda]_+ \quad \text{s.t. } q_i \in [0, 1], \ ||q||_0 = n - m,$$

where $qs(\theta) = \{q_1 s_1(\theta), \cdots, q_n s_n(\theta)\}$. The first equation holds because of Lemma A.1. Since $q_i s_i(\theta) \geq 0$, we know the optimal $\lambda^* \geq 0$ from Lemma A.1. If $q_i = 0$, $[q_i s_i(\theta) - \lambda^*]_+ = 0 = q_i[s_i(\theta) - \lambda^*]_+$. If $q_i = 1$, $[q_i s_i(\theta) - \lambda^*]_+ = [s_i(\theta) - \lambda^*]_+ = q_i[s_i(\theta) - \lambda^*]_+$. Thus the second equation holds. It should be mentioned that the discrete indicator q_i can be replaced by a continue one, which means $q_i \in [0, 1]$. The reason is that the optimal q is at a corner of the hypercube $[0, 1]^n$. In fact, with the outer minimization of model parameter θ, the above objective function is equivalent to equation (1). Specifically, we obtain a theorem as follows,

Theorem 2 *With relaxing q to $[0, 1]$, equation (4) is equivalent to equation (1):*

$$\min_{\lambda, q}(k - m)\lambda + \sum_{i=1}^{n} q_i [s_i(\theta) - \lambda]_+ \quad s.t. \ q_i \in [0, 1], \ ||q||_0 = n - m$$

$$= \min_{\lambda} \left\{ k\lambda + \sum_{i=1}^{n} [s_i(\theta) - \lambda]_+ \right\} - \min_{\hat{\lambda}} \left\{ m\hat{\lambda} + \sum_{i=1}^{n} [s_i(\theta) - \hat{\lambda}]_+ \right\}.$$

Proof can be found in Appendix A.3. Combining the optimization of the model parameter θ, we can reform equation (4) to a bilevel optimization problem (Borsos et al., 2020; Jenni and Favaro, 2018) as follows,

$$\min_{\lambda, q} \quad (k - m)\lambda + \sum_{i=1}^{n} q_i [s_i(\theta^*(q)) - \lambda]_+$$

$$\text{s.t.} \quad \theta^*(q) \in \underset{\theta}{\operatorname{argmin}}(k - m)\lambda + \sum_{i=1}^{n} q_i [s_i(\theta) - \lambda]_+ \quad (5)$$

$$q_i \in [0, 1], \ ||q||_0 = n - m,$$

where we minimize an *outer* objective, here the first row of equation (5), which in turn depends on the solution $\theta^*(q)$ to an *inner* optimization problem (the second row of equation (5)) with the constraint on q. This problem (equation (5)) can be solved through some existing bilevel optimization algorithms (Borsos et al., 2020). For example, we can adopt the *coresets via bilevel optimization* algorithm from Borsos et al. (2020) to solve our cardinality-constrained bilevel optimization problem. We only need to take our specific inner problem, outer problem, and constraints into their optimization framework.

4. AoRR Aggregate Loss for Binary and Multi-class Classification

SoRR provides a general framework to aggregate individual values to form a learning objective. Here, we examine its use as an aggregate loss in supervised learning problems in detail and optimize it using the DC algorithm. Specifically, we aim to find a parametric function f_θ with parameter θ that can predict a target y from the input data or features x using a set of labeled training samples $\{(x_i, y_i)\}_{i=1}^n$. We assume that the individual loss for a sample (x_i, y_i) is $s_i(\theta) = s(f(x_i; \theta), y_i) \geq 0$. The learning objective for supervised learning problem is constructed from the aggregate loss $\mathcal{L}(S(\theta))$ that accumulates all individual losses over training samples, $S(\theta) = \{s_i(\theta)\}_{i=1}^n$. Specifically, we define the AoRR aggregate loss as

$$\mathcal{L}_{aorr}(S(\theta)) = \frac{1}{k-m}\psi_{m,k}(S(\theta)) = \frac{1}{k-m}\sum_{i=m+1}^{k} s_{[i]}(\theta).$$

If we choose the ℓ_2 individual loss or the hinge individual loss, we get the learning objectives in Ortis et al. (2019) and Kanamori et al. (2017), respectively. For $m \geq 1$, we can optimize AoRR using the DCA as described in Section 3.

The AoRR aggregate loss is related to previous aggregate losses that are widely used to form learning objectives.

- the *average loss* (Vapnik, 2013): $\mathcal{L}_{avg}(S(\theta)) = \frac{1}{n}\sum_{i=1}^n s_i(\theta)$;
- the *maximum loss* (Shalev-Shwartz and Wexler, 2016): $\mathcal{L}_{max}(S(\theta)) = \max_{1 \leq i \leq n} s_i(\theta)$;
- the *median loss* (Ma et al., 2011): $\mathcal{L}_{med}(S(\theta)) = \frac{1}{2}\left(s_{\left[\lfloor \frac{n+1}{2}\rfloor\right]}(\theta) + s_{\left[\lceil \frac{n+1}{2}\rceil\right]}(\theta)\right)$;
- the *average top-k loss* (AT$_k$) (Fan et al., 2017): $\mathcal{L}_{avt-k}(S(\theta)) = \frac{1}{k}\sum_{i=1}^k s_{[i]}(\theta)$, for $1 \leq k \leq n$.

The AoRR aggregate loss generalizes the average loss ($k = n$ and $m = 0$), the maximum loss ($k = 1$ and $m = 0$), the median loss ($k = \lceil \frac{n+1}{2}\rceil$, $m = \lfloor \frac{n+1}{2}\rfloor - 1$) , and the average top-k loss ($m = 0$). Interestingly, the average of the bottom-$(n-m)$ loss, $\mathcal{L}_{abt-m}(S(\theta)) = \frac{1}{n-m}\sum_{i=m+1}^n s_{[i]}(\theta)$, which is not widely studied in the literature as a learning objective, is an instance of the AoRR aggregate loss ($k = n$). In addition, the robust version of the maximum loss (Shalev-Shwartz and Wexler, 2016), which is a maximum loss on a subset of samples of size at least $n - (k-1)$, where the number of outliers is at most $k-1$, is equivalent to the top-k loss, a special case of the AoRR aggregate loss ($m = k - 1$).

Using the AoRR aggregate loss can bring flexibility in designing learning objectives and alleviate drawbacks of previous aggregate losses. In particular, the average loss, the maximum loss, and the AT$_k$ loss are all influenced by outliers in training data since outliers usually correspond to extremely large individual losses and all of these methods inevitably use large individual losses to construct aggregate losses. They only differ in the degree of influence, with the maximum loss being the most sensitive to outliers. In comparison, AoRR loss can completely eliminate the influence of the top individual losses by excluding the top m individual losses from the learning objective.

In addition, traditional approaches to handling outliers focus on the design of robust *individual losses* over training samples, notable examples include the Huber loss (Friedman et al., 2001) and the capped hinge loss (Nie et al., 2017). Changing individual losses may not be desirable, as they are usually relevant to the learning problem and application. On the other hand, instead of changing the definition of individual losses, our method builds in
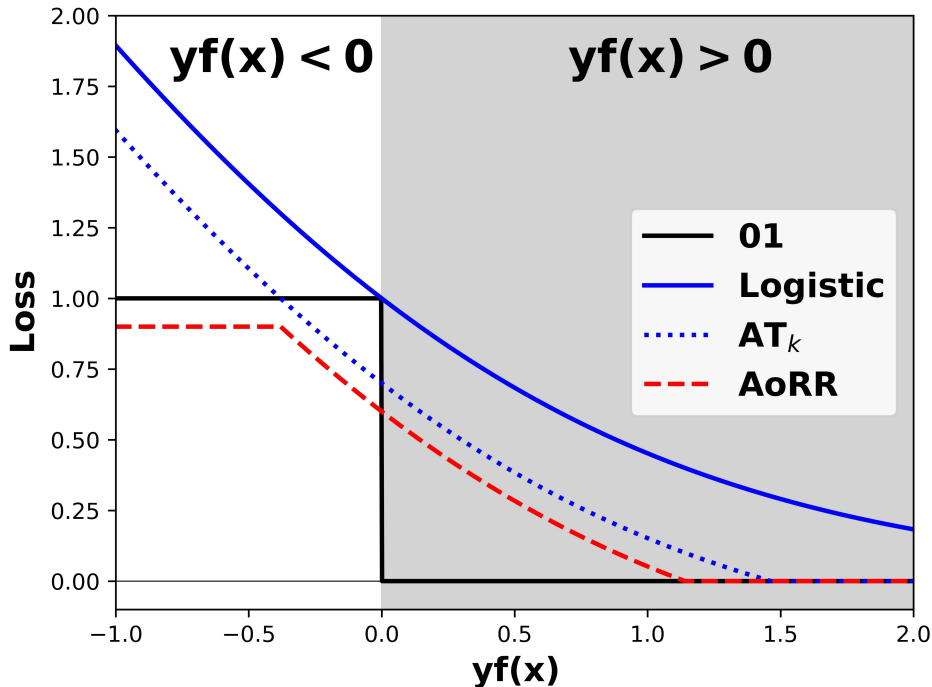
Figure 2: The `AoRR` loss and other losses interpreted at the individual sample level. The shaded area over 0.00 loss corresponds to data/target with the correct classification. Note that we fix $\lambda = 0.4$ and $\hat{\lambda} = 1.3$ to draw the `AoRR` curve.

robustness at the aggregate loss level, using the `AoRR` aggregate loss. The resulting learning algorithm based on Algorithm 1 is more flexible and allows the user to choose an individual loss form that is relevant to the learning problem.

The robustness to outliers of the `AoRR` loss can be more clearly understood at the individual sample level, with fixed λ and $\hat{\lambda}$. We use binary classification to illustrate with $s_i(\theta) = s(y_i f_\theta(x_i))$ where f_θ is the parametric predictor and $y_i \in \{\pm 1\}$. In this case, $y_i f_\theta(x_i) > 0$ and $y_i f_\theta(x_i) < 0$ corresponds to the correct and false predictions, respectively. Specifically, noting that $s_i(\theta) \geq 0$, we can rearrange terms in equation (1) to obtain

$$\mathcal{L}_{aorr}(S(\theta)) = \frac{1}{k-m} \min_{\lambda > 0} \max_{\hat{\lambda} > \lambda} \sum_{i=1}^{n} \left\{ [s_i(\theta) - \lambda]_+ - [s_i(\theta) - \hat{\lambda}]_+ \right\} + k\lambda - m\hat{\lambda}. \qquad (6)$$

We are particularly interested in the term inside the summation in equation (6)

$$[s(yf_\theta(x)) - \lambda]_+ - [s(yf_\theta(x)) - \hat{\lambda}]_+ = \begin{cases} \hat{\lambda} - \lambda & s(yf_\theta(x)) > \hat{\lambda} \\ s(yf_\theta(x)) - \lambda & \lambda < s(yf_\theta(x)) \leq \hat{\lambda} \\ 0 & s(yf_\theta(x)) \leq \lambda \end{cases}.$$

According to this, at the level of individual training samples, the equivalent effect of using the `AoRR` loss is to uniformly reduce the individual losses by λ, but truncate the reduced individual loss at values below zero or above $\hat{\lambda}$. The situation is illustrated in Figure 2 for the logistic individual loss $s(yf(x)) = \log_2(1 + e^{-yf(x)})$, which is a convex and smooth

surrogate to the ideal 01-loss. The effect of reducing and truncating from below and above has two interesting consequences. First, note that the use of convex and smooth surrogate loss inevitably introduces penalties to samples that are correctly classified but are "too close" to the boundary. The reduction of the individual loss alleviates that improper penalty. This property is also shared by the AT_k loss. On the other hand, the ideal 01-loss exerts the same penalty to all incorrect classified samples regardless of their margin value, while the surrogate has unbounded penalties. This is the exact cause of the sensitivity to outliers of the previous aggregate losses, but the truncation of `AoRR` loss is similar to the 01-loss, and thus is more robust to the outliers. It is worth emphasizing that the above explanation of the `AoRR` loss has been illustrated at the individual sample level with fixed λ and $\hat{\lambda}$. The aggregate `AoRR` loss defined by equation (6) as a whole is not an average sample-based loss because it can not be decomposed into the summation of individual losses over samples.

4.1 Classification Calibration

A fundamental question in learning theory for classification (Bartlett et al., 2006; Vapnik, 2013) is to investigate when the best possible estimator from a learning objective is consistent with the best possible, *i.e.*, the Bayes rule. Here we investigate this statistical question for the `AoRR` loss by considering its infinite sample case, *i.e.*, $n \to \infty$. As mentioned above, the `AoRR` loss as a whole is not the average of individual losses over samples, and therefore the analysis for the standard ERM (Bartlett et al., 2006; Lin, 2004) does not apply to our case.

We assume that the training data $\{(x_i, y_i)\}_{i=1}^n$ are i.i.d. from an unknown distribution p on $\mathcal{X} \times \{\pm 1\}$. The misclassification error measures the quality of a classifier $f : \mathcal{X} \to \{\pm 1\}$ and is denoted by $\mathcal{R}(f) = \Pr(Y \neq f(X)) = \mathbb{E}[\mathbb{I}_{Yf(X) \leq 0}]$. The Bayes error leads to the least expected error, which is defined by $\mathcal{R}^* = \inf_f \mathcal{R}(f)$. No function can achieve the Bayes error than the Bayes rule $f_c(x) = \text{sign}(\eta(x) - \frac{1}{2})$, where $\eta(x) = P(Y = 1|X = x)$. It is well noted that, in practice, one uses a surrogate loss $\ell : \mathbb{R} \to [0, \infty)$ which is a continuous function and upper-bounds the 01-loss. Its true risk is given by $\mathcal{E}_\ell(f) = \mathbb{E}[\ell(Yf(X))]$. Denote the optimal ℓ-risk by $\mathcal{E}_\ell^* = \inf_f \mathcal{E}_\ell(f)$, the *classification calibration* (point-wise form of Fisher consistency) for loss ℓ (Bartlett et al., 2006; Lin, 2004) holds true if the minimizer $f_\ell^* = \inf_f \mathcal{E}_\ell(f)$ has the same sign as the Bayes rule $f_c(x)$, *i.e.*, $\text{sign}(f_\ell^*(x)) = \text{sign}(f_c(x))$ whenever $f_c(x) \neq 0$.

In analogy, we can investigate the classification calibration property of the `AoRR` loss. Specifically, we first obtain the population form of the `AoRR` loss using the infinite limit of the empirical one given by equation (6). Indeed, we know from (Bhat and Prashanth, 2019; Brown, 2007) that, for any bounded f and $\alpha \in (0, 1]$, there holds $\inf_{\lambda \geq 0} \alpha\lambda + \frac{1}{n}\sum_{i=1}^n [s(y_i f_\theta(x_i)) - \lambda]_+ \to \inf_{\lambda \geq 0} \alpha\lambda + \mathbb{E}[s(Yf(X)) - \lambda]_+$ as $n \to \infty$. Consequently, we have the limit case of the `AoRR` loss $\mathcal{L}_{aorr}(S(\theta))$ restated as follows:

$$
\frac{n}{k-m}\left[\min_\lambda \left\{\frac{k}{n}\lambda + \frac{1}{n}\sum_{i=1}^n [s(y_i f_\theta(x_i)) - \lambda]_+\right\} - \min_{\hat{\lambda}} \left\{\frac{m}{n}\hat{\lambda} + \frac{1}{n}\sum_{i=1}^n [s(y_i f_\theta(x_i)) - \hat{\lambda}]_+\right\}\right]
$$
$$
\xrightarrow[n\to\infty]{\frac{k}{n}\to\nu, \frac{m}{n}\to\mu} \frac{n}{k-m}\left[\min_{\lambda\geq 0} \left\{\mathbb{E}[[s(Yf(X)) - \lambda]_+] + \nu\lambda\right\} - \min_{\hat{\lambda}\geq 0} \left\{\mathbb{E}[[s(Yf(X)) - \hat{\lambda}]_+] + \mu\hat{\lambda}\right\}\right].
$$
(7)

Throughout the paper, we assume that $\nu > \mu$ which is reasonable as $k > m$. In particular, we assume that $\mu > 0$ since if $\mu = 0$ then it will lead to $\hat{\lambda} = \infty$ and this case is reduced to the population version of the average top-k case in Fan et al. (2017). As such, the population

version of our `AoRR` loss (equation (6)) is given by

$$(f_0^*, \lambda^*, \hat{\lambda}^*) = \arg \inf_{f, \lambda \geq 0} \sup_{\hat{\lambda} \geq 0} \left\{ \mathbb{E}[[s(Yf(X)) - \lambda]_+ - [s(Yf(X)) - \hat{\lambda}]_+] + (\nu\lambda - \mu\hat{\lambda}) \right\}. \quad (8)$$

It is difficult to directly work on the optima f_0^* since the problem in equation (8) is a non-convex min-max problem and the standard min-max theorem does not apply here. Instead, we assume the existence of λ^* and $\hat{\lambda}^*$ in equation (8) and work with the minimizer $f^* = \arg \inf_f \mathcal{L}(f, \lambda^*, \hat{\lambda}^*)$ where $\mathcal{L}(f, \lambda^*, \hat{\lambda}^*) := \mathbb{E}[[s(Yf(X)) - \lambda^*]_+ - [s(Yf(X)) - \hat{\lambda}^*]_+] + (\nu\lambda^* - \mu\hat{\lambda}^*)$. Now we can define the classification calibration for the `AoRR` loss.

Definition 3 *The `AoRR` loss is called classification calibrated if, for any x, there is a minimizer $f^* = \arg \inf_f \mathcal{L}(f, \lambda^*, \hat{\lambda}^*)$ such as $f^*(x) > 0$ if $\eta(x) > 1/2$ and $f^*(x) < 0$ if $\eta(x) < 1/2$.*

We can then obtain the following theorem. Its proof can be found in Appendix A.4.

Theorem 4 *Suppose the individual loss $s : \mathbb{R} \to \mathbb{R}^+$ is non-increasing, convex, differentiable at 0 and $s'(0) < 0$. If $0 \leq \lambda^* < \hat{\lambda}^*$, then the `AoRR` loss is classification calibrated.*

The assumptions in Theorem 4 are easily to be satisfied. Indeed, the commonly used individual losses such as the hinge loss $s(t) = [1 - t]_+$ and the logistic loss $s(t) = \log_2(1 + e^{-t})$ satisfy the conditions of non-increasing, convex, differentiable at 0, and $s'(0) < 0$. Furthermore, according to Theorem 1 and $0 \leq m < k \leq n$, we obtain $\lambda^* < \hat{\lambda}^*$. Suppose the individual loss $s(t)$ is non-negative (this holds for the above-mentioned individual losses), we have $0 \leq \lambda^* < \hat{\lambda}^*$. We also provide several examples that cannot make the assumptions hold. For example, the least square loss $s(t) = (1 - t)^2$ is not a non-increasing individual loss. Therefore, using it as an individual loss cannot guarantee the `AoRR` aggregate loss satisfies classification calibration. In addition, if $k \leq m$, the hypothesis $0 \leq \lambda^* < \hat{\lambda}^*$ will not be held. So we also cannot obtain a classification calibrated `AoRR` aggregate loss.

4.2 Connection with Conditional Value at Risk

It is worthy of mentioning that the average of the top-k method is also related to the risk measure called conditional value at risk (CVaR) at level $\alpha = \frac{k}{n}$ (Shapiro et al., 2014, Chapter 6) in portfolio optimization for effective risk management.

Let (Ω, Σ, P) be a probability space. The *conditional value at risk* (CVaR) at level $\alpha \in (0, 1)$ of a random variable $s : \Omega \to \mathbb{R}$ is defined as $C_\alpha[s] := \inf_\lambda \{\lambda + \frac{1}{\alpha}\mathbb{E}[[s - \lambda]_+]\}$. If s is a continuous random variable, then $C_\alpha[s] = \mathbb{E}[s | s \geq V_\alpha[s]]$, where $V_\alpha[s]$ is called the *value at risk* (VaR) and defined as $V_\alpha[s] := \sup\{\lambda \in \mathbb{R} | \Pr(s \geq \lambda) \geq \alpha\}$. A sample-based estimate of $C_\alpha[s]$ can be denoted by $\widehat{C}_\alpha[s] := \inf_{\lambda \in \mathbb{R}}\{\lambda + \frac{1}{n\alpha}\sum_{i=1}^n[s_i - \lambda]_+\}$, which is also called the empirical of $C_\alpha[s]$. The CVaR has a natural distributionally robust optimization interpretation (Shapiro et al., 2014). Therefore, from equation (8), we know the population version of `AoRR` $\mathcal{L}(f, \lambda, \hat{\lambda}) = \inf_\lambda \sup_{\hat{\lambda}} \left\{ \mathbb{E}[[s(Yf(X)) - \lambda]_+ - [s(Yf(X)) - \hat{\lambda}]_+] + (\nu\lambda - \mu\hat{\lambda}) \right\}$ can be expressed by difference of two CVaRs. We call it as Interval Conditional Value at Risks (ICVaRs).

$$\mathcal{L}(f, \lambda, \hat{\lambda}) = \nu C_\nu[s(\theta)] - \mu C_\mu[s(\theta)]. \quad (9)$$
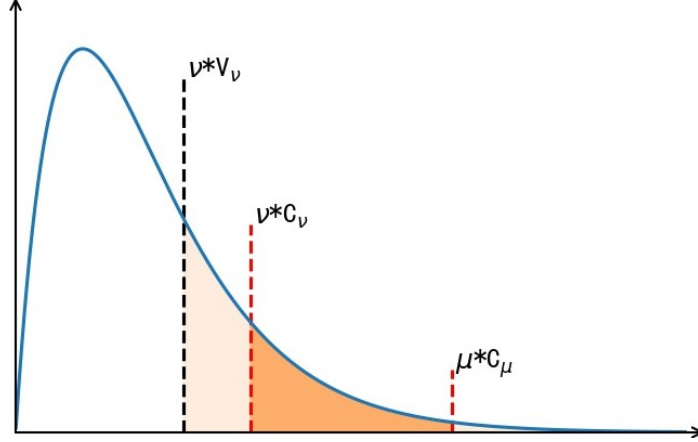
Figure 3: The yellow area between two red dash lines represents ICVaRs

Figure 3 is an interpretation of ICVaRs. We denote the empirical form of $\mathcal{L}(f, \lambda, \hat{\lambda})$ by $\widehat{\mathcal{L}}(f, \lambda, \hat{\lambda})$, and

$$\widehat{\mathcal{L}}(f, \lambda, \hat{\lambda}) = \nu \widehat{C}_\nu[s(\theta)] - \mu \widehat{C}_\mu[s(\theta)]. \tag{10}$$

The following theorem provides some deviation convergence bounds for estimating $\psi_{m,k}(S(\theta))$ from a finite number of independent samples.

Theorem 5 *If $supp(s(\theta)) \subseteq [a, b]$ and s has a continuous distribution function, then for any $\delta \in (0, 1]$,*

$$\Pr\left(\mathcal{L}(f, \lambda, \hat{\lambda}) - \widehat{\mathcal{L}}(f, \lambda, \hat{\lambda}) \leq (b - a)\left[\sqrt{\frac{5k\ln(3/\delta)}{n^2}} + \sqrt{\frac{\ln(1/\delta)}{2n}}\right]\right) \geq 1 - \delta,$$

$$\Pr\left(\mathcal{L}(f, \lambda, \hat{\lambda}) - \widehat{\mathcal{L}}(f, \lambda, \hat{\lambda}) \geq -(b - a)\left[\sqrt{\frac{5m\ln(3/\delta)}{n^2}} + \sqrt{\frac{\ln(1/\delta)}{2n}}\right]\right) \geq 1 - \delta.$$

Proof can be found in Appendix A.5. It should be mentioned that we can derive better generalization bounds in some mild conditions (sub-Gaussian, light-tailed, and heavy-tailed data distributions) according to Prashanth et al. (2020); Thomas and Learned-Miller (2019). Note that the non-asymptotic relation between the excess generalization induced by `AoRR` and the target classification problem (e.g. excess misclassification error) is a fundamentally important question. However, it is very hard to establish this relation since the `AoRR` involves that the difference of two possible convex losses, i.e. $[s(yf_\theta(x)) - \lambda]_+ - [s(yf_\theta(x)) - \hat{\lambda}]_+$. We leave this relation for future study.

4.3 Determine k and m

Hyper-parameters k and m are very important for the final performance of the `AoRR` aggregate loss. However, before actually training the model, it is difficult to determine their suitable values so that the model can achieve the best performance. We cannot fix them to some specific values. For different data sets, they may have different suitable values. Greedy search is a good method for us to find the optimal k and m in a finite time, but it only works on

simple data sets that contain a small number of samples. For large data sets, the greedy search method is not efficient and could be very time-consuming.

In practice, for large-scale data sets, we can decide on an approximate value of m if we have prior knowledge about the faction of outliers in the data set. To avoid extra freedom due to the value of k, we follow a very popular adaptive setting that has been applied in previous works (e.g., Kawaguchi and Lu (2020)). At the beginning of training, k equals to the size (n) of training data, $k = \lfloor \frac{n}{2} \rfloor$ once training accuracy $\geq 70\%$, $k = \lfloor \frac{n}{4} \rfloor$ once training accuracy $\geq 80\%$, $k = \lfloor \frac{n}{8} \rfloor$ once training accuracy $\geq 90\%$, $k = \lfloor \frac{n}{16} \rfloor$ once training accuracy $\geq 95\%$, $k = \lfloor \frac{n}{32} \rfloor$ once training accuracy $\geq 99.5\%$. However, if we do not have prior knowledge about the fraction of outliers in the data set, the adaptive setting method still needs to try different values of m from its feasible space and then find the optimal values of k and m. Obviously, the time complexity is dominated by the size of m's feasible set. Therefore, this method will take a long time to search for the optimal hyper-parameters if we apply it to a large data set.

Many existing works on label corruption or label noise problem assume that training data is not clean and potentially includes noise and outliers. However, in general, there are many trusted samples are available. These trusted data can be collected to create a clean validation set. This assumption has been analyzed in Charikar et al. (2017) and also been used by other works for designing a robust learning model (Hendrycks et al., 2018; Ren et al., 2018; Veit et al., 2017; Li et al., 2017b). Hendrycks et al. (2018) proposes a loss correction technique for deep neural network classifiers that use clean data to alleviate the influences of label noise. Li et al. (2017b) proposes a distillation framework by using a clean data set to reduce the risk of learning from noisy labels. We apply this assumption to our problem. Specifically, we extract a clean validation set from training data to determine the values of k and m. According to Theorem 1, fix θ, we will get the optimal values of λ and $\hat{\lambda}$ if we know the values of k and m. Based on the values of λ and $\hat{\lambda}$, we can calculate how many individual losses are larger than them. These also provide us information about k and m. Therefore, we can directly calculate the optimal λ and $\hat{\lambda}$ based on the validation set instead of determining the optimal values of k and m.

According to the above analysis, we propose a framework (Figure 4) to combine hyper-parameters learning and Algorithm 1. To better learn the parameters of λ and $\hat{\lambda}$, we first use the average aggregate loss to "warm-up" the model for some epochs by training on the training data $\mathcal{D}$, which contains outliers. This pre-processing has been used in many existing works such as Li et al. (2020a); Hendrycks et al. (2018). As shown in Figure 4, to "warm-up" the model, one needs to use forward and backward (see Steps 1 and 2) to update the model parameter $\theta^{(t)}$ with learning rate η_t. After a few epochs, with a stopping criterion (Step 3), we use model $\theta^{(t)}$ on the clean validation set $\widetilde{\mathcal{D}}$ to learn the individual losses (Step 4). Through simple operations 1 and 2, we can get $\hat{\lambda}$ and λ. For example, we can calculate the mean value plus one standard deviation of all individual losses from $\widetilde{\mathcal{D}}$ and then regard it as operation 1 and regard the mean value minus two standard deviations of all individual losses from $\widetilde{\mathcal{D}}$ as operation 2. Note that we should keep $\hat{\lambda} > \lambda$ according to Theorem 1. When we have $\hat{\lambda}$, we can obtain sub-gradient $\hat{\theta}$ based on training data $\mathcal{D}$ and equation (3). With $\mathcal{D}, \hat{\theta}$ and λ, we use an inner loop and stochastic gradient descent method with learning rate η_l to update the inner convex model parameter $\theta^{(l)}$. Meanwhile, apply $\theta^{(l)}$ on $\widetilde{\mathcal{D}}$ and re-use operation 2 to update the parameter λ. This procedure (Steps 5, 6, and 7) is similar to equation (2). With another stopping criterion (Step 8), we can update $\theta^{(t)}$ by using $\theta^{(l)}$ and
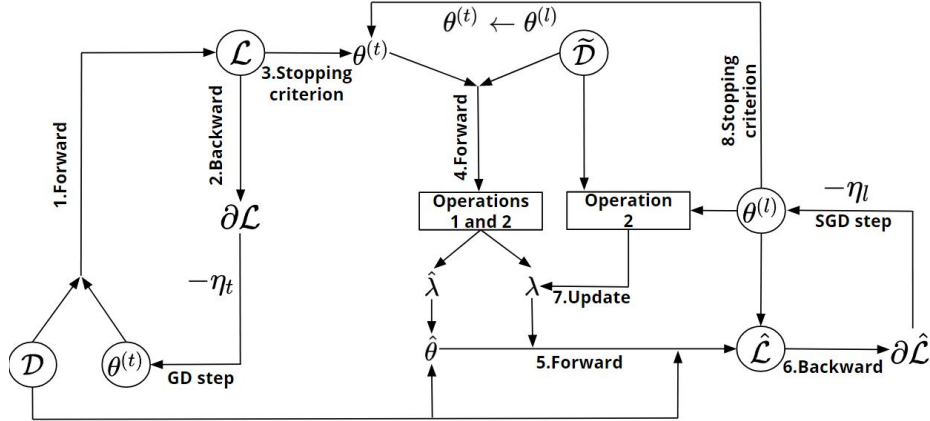
Figure 4: The framework of learning hyper-parameters. Note that we calculate the mean value plus one standard deviation of all individual losses from $\widetilde{\mathcal{D}}$ and then regard it as operation 1 and regard the mean value minus two standard deviations of all individual losses from $\widetilde{\mathcal{D}}$ as operation 2 in practice.

redo steps 4, 5, 6, 7, and 8 repeatedly until the final stopping criterion is satisfied. We list detailed step-by-step pseudo-code in Algorithm 2. For a given outer loop size $|t|$, an inner loop size $|l|$, the "warm-up" loop size $|p|$, training sample size n, and the validation sample size $\widetilde{n}$, the time complexity of Algorithm 2 is $O((|p| + |t|)n + |t| \cdot |l| \cdot \widetilde{n})$.

4.4 Experiments

We empirically demonstrate the effectiveness of the AoRR aggregate loss combined with two types of individual losses for binary classification, namely, the logistic loss and the hinge loss. For simplicity, we consider a linear prediction function $f(x; \theta) = \theta^T x$ with parameter θ, and the ℓ_2 regularizer $\frac{1}{2C}||\theta||_2^2$ with $C > 0$. For binary classification, we apply the greedy search method to select optimal hyper-parameters k and m. We use the MNIST data set with symmetric (i.e. uniformly random) label noise to verify the effectiveness of our Algorithm 2 for learning hyper-parameters and show the AoRR aggregate loss can also be extended to multi-class classification. All algorithms are implemented in Python 3.6 and trained and tested on an Intel(R) Xeon(R) CPU W5590 @3.33GHz with 48GB of RAM.

4.4.1 AoRR for Binary Classification

Synthetic data. We generate two sets of 2D synthetic data (Figure 5). Each data set contains 200 samples from Gaussian distributions with different means and variances. We consider both the case of the balanced (Figure 5 (a,b)) and the imbalanced (Figure 5 (c,d)) data distributions, in the former the training data for the two classes are approximately equal while in the latter one class has a dominating number of samples in comparison to the other. The learned linear classifiers with different aggregate losses are shown in Figure 5. Both data sets have an outlier in the blue class (shown as ×).

To optimally remove the effect of outliers, we need to set k larger than the number of outliers in the training data set. Since there is one outlier in this synthetic data set, we

---

Algorithm 2: DCA for Minimizing `AoRR` without Setting k and m

---

1 **Initialization:** $\theta^{(0)}$, η_t, η_l, $\hat{\lambda} = 0$, $\lambda = 0$, flag=0, training data set $\mathcal{D}$ (with outliers), and validation data set $\widetilde{\mathcal{D}}$ (without outliers).

2 **for** $t = 0, 1, ...$ **do**

3 $\quad$ **if** *flag == 0* **then**

4 $\quad\quad$ **for** $p = 0, 1, ...$ **do**

5 $\quad\quad\quad$ $\theta^{(p+1)} \leftarrow \theta^{(p)} - \frac{\eta_t}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \partial s_i(\theta)$

6 $\quad\quad\quad$ **if** *Stopping criterion* **then**

7 $\quad\quad\quad\quad$ $\hat{\lambda} \leftarrow \text{operation1}(\{s_j(\theta^{(p+1)})|j \in \widetilde{\mathcal{D}}\})$

8 $\quad\quad\quad\quad$ $\lambda \leftarrow \text{operation2}(\{s_j(\theta^{(p+1)})|j \in \widetilde{\mathcal{D}}\})$

9 $\quad\quad\quad\quad$ flag $\leftarrow 1$, $\theta^{(0)} \leftarrow \theta^{(p+1)}$

10 $\quad\quad\quad\quad$ **break**

11 $\quad\quad\quad$ **end**

12 $\quad\quad$ **end**

13 $\quad$ **else**

14 $\quad\quad$ Compute $\hat{\theta}^{(t)} \in \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \partial s_i(\theta^{(t)}) \cdot \mathbb{I}_{[s_i(\theta^{(t)}) > \hat{\lambda}]}$

15 $\quad\quad$ **for** $l = 0, 1, ...$ **do**

16 $\quad\quad\quad$ Randomly sample $s_{i_l}(\theta^{(l)})$ from the collection of $\{s_i(\theta^{(l)})\}_{i \in \mathcal{D}}$

17 $\quad\quad\quad$ $\theta^{(l+1)} \leftarrow \theta^{(l)} - \eta_l \left(\partial s_{i_l}(\theta^{(l)}) \cdot \mathbb{I}_{[s_{i_l}(\theta^{(l)}) > \lambda]} - \hat{\theta}^{(t)} \right)$

18 $\quad\quad\quad$ $\lambda \leftarrow \text{operation2}(\{s_j(\theta^{(l+1)})|j \in \widetilde{\mathcal{D}}\})$

19 $\quad\quad$ **end**

20 $\quad\quad$ Update $\theta^{(t+1)} \leftarrow \theta^{(l+1)}$

21 $\quad\quad$ $\hat{\lambda} \leftarrow \text{operation1}(\{s_j(\theta^{(t+1)})|j \in \widetilde{\mathcal{D}}\})$, $\lambda \leftarrow \text{operation2}(\{s_j(\theta^{(t+1)})|j \in \widetilde{\mathcal{D}}\})$

22 $\quad$ **end**

23 **end**

---

select $k = 2$ here as an example. As shown in Figure 5, neither the maximum loss nor the average loss performs well on the synthetic data set, due to the existence of outliers and the multi-modal nature of the data. Furthermore, Figure 5 also shows that the AT_k loss does not bode well: it is still affected by outliers. The reason can be that the training process with the AT_k loss with $k = 2$ will most likely pick up one individual loss from the outlier for optimization. In contrast, the `AoRR` loss with $k=2$ and $m=1$, which is equivalent to the top-2 or second-largest individual loss, yields better classification results. Intuitively, we avoid the direct effects of the outlier since it has the largest individual loss value. Furthermore, we perform experiments to show misclassification rates of `AoRR` with respect to different values of k in Figure 5 (e), (f), (g), (h) for each case and compare with the AT_k loss and optimal Bayes classifier. The results show that for k values other than 2, the `AoRR` loss still exhibits an advantage over the AT_k loss. Our experiments are based on a grid search for selecting the value of k and m because we found it is simple and often yields comparable performance.
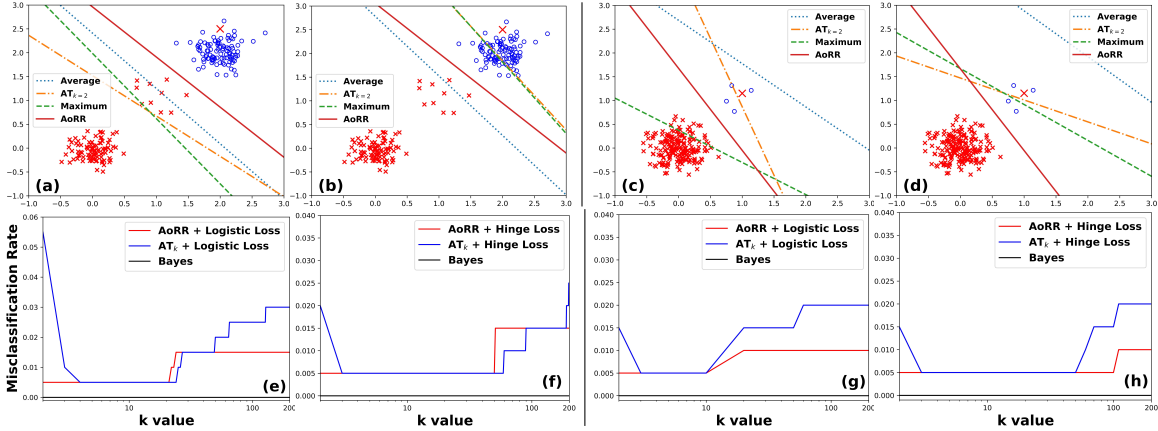
Figure 5: Comparison of different aggregate losses for binary classification on a balanced but multi-modal synthetic data set and with outliers with logistic loss (a) and hinge loss (b), and an imbalanced synthetic data set with outliers with logistic loss (c) and hinge loss (d). Outliers in data are shown as × in blue class. The figures (e), (f), (g) and (h) show the misclassification rates of AoRR w.r.t. different values of k for each case and compare with the AT_k and the optimal Bayes classifier.

In order to evaluate the effects of different aggregate losses on more than one outlier, we also conduct additional experiments on the multi-modal toy example that includes more outliers. We consider six cases as follows,

- **Case 1 (2 outliers).** In Figure 6 (a) and (b), there exist two outliers. Let hyper-parameters $k = 3$ and $m = 2$.
- **Case 2 (3 outliers).** Figure 6 (c) and (d) contain three outliers. In this scenario, $k = 4$ and $m = 3$.
- **Case 3 (4 outliers).** Figure 6 (e) and (f) include four outliers and we set $k = 5$ and $m = 4$.
- **Case 4 (5 outliers).** There are five outliers in Figure 6 (g) and (h). We set $k = 6$ and $m = 5$.
- **Case 5 (10 outliers).** Ten outliers have been included in Figure 6 (i) and (j). Let $k = 11$ and $m = 10$ in this case.
- **Case 6 (20 outliers).** We create twenty outliers in Figure 6 (k) and (l) and make $k = 21$ and $m = 20$.

Seeing on cases 1, 2, 3, and 4, the linear classifier learned from average aggregate loss cross some red samples from minor distribution even though the data is separable. The reason is that the samples close to the decision boundary are sacrificed to reduce the total loss over the whole data set. The AT_k loss selects k largest individual losses which contain many outliers to train the classifier. It leads to the instability of the learned classifier. This phenomenon can be found when we compare all cases. Similarly, the maximum aggregate loss cannot fit this data very well in all cases. This loss is very sensitive to outliers.

In cases 5 and 6, the average aggregate loss with individual logistic loss achieves better results than with individual hinge loss. A possible reason is that for correctly classified samples with a margin greater than 1, the penalty caused by hinge loss is 0. However, it is
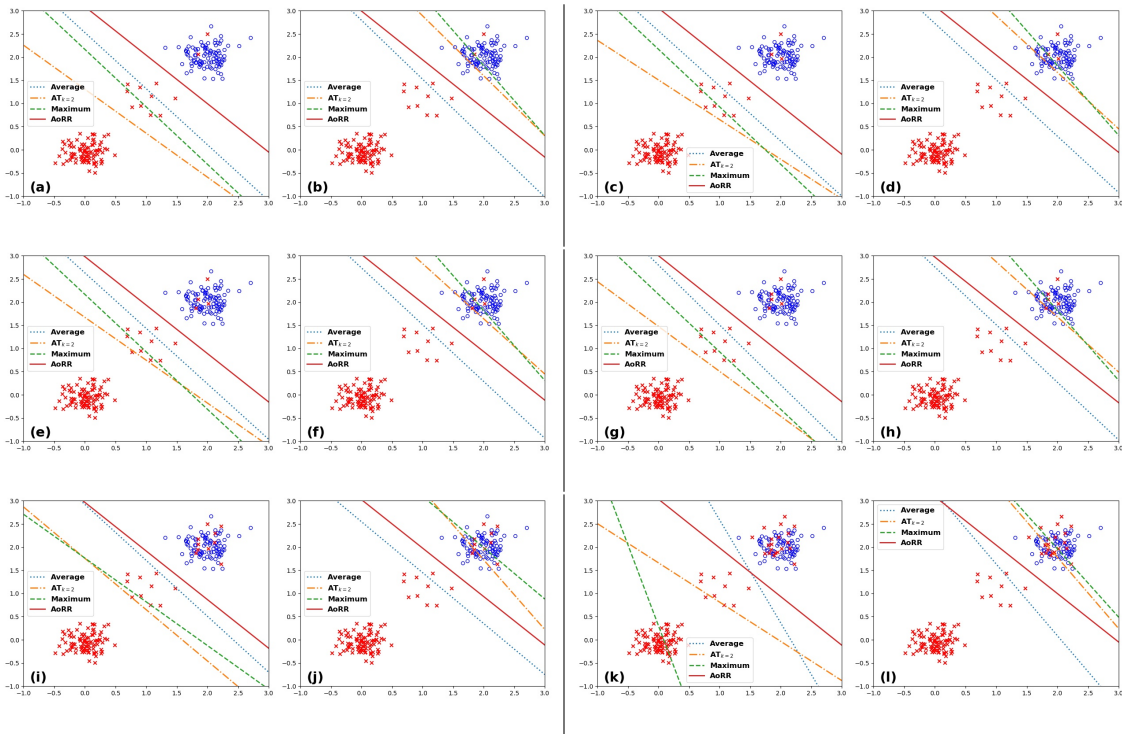
Figure 6: Comparison of different aggregate losses on 2D synthetic data with 200 samples for binary classification with individual logistic loss (a, c, e, g, i, k) and individual hinge loss (b, d, f, h, j, l). Outliers are shown as × in blue class.

Data Sets	#Classes	#Samples	#Features	Class Ratio
Monk	2	432	6	1.12
Australian	2	690	14	1.25
Phoneme	2	5,404	5	2.41
Titanic	2	2,201	3	2.10
Splice	2	3,175	60	1.08

Table 2: Statistical information of five real data sets.

non-zero when using logistic loss. Since many outliers in the blue class, to reduce the average loss, the decision boundary will close to the blue class. Especially, when we compare (i) and (k), it is obvious that average loss can achieve a better result while the number of outliers is increasing.

As we discussed, the hinge loss has less penalty for correctly classified samples than logistic loss. This causes outliers to be more prominent than normal samples while using the individual hinge loss. This result can be verified in the experiment when we compare the individual logistic loss and the individual hinge loss. For example, (i) and (j), (k) and (l), etc. We find the decision boundaries of maximum loss and AT_k loss are close to outliers in the individual hinge loss scenario because both of them are sensitive to outliers in our cases.

Real data. We use five benchmark data sets from the UCI (Dua and Graff, 2017) and the KEEL (Alcalá-Fdez et al., 2011) data repositories (statistical information of each data set

Data Sets	Logistic Loss					Hinge Loss				
	Maximum	R_Max	Average	AT_k	AoRR	Maximum	R_Max	Average	AT_k	AoRR
Monk	22.41	21.69	20.46	16.76	**12.69**	22.04	20.61	18.61	17.04	**13.17**
	(2.95)	(2.62)	(2.02)	(2.29)	**(2.34)**	(3.08)	(3.38)	(3.16)	(2.77)	**(2.13)**
Australian	19.88	17.65	14.27	11.7	**11.42**	19.82	15.88	14.74	12.51	**12.5**
	(6.64)	(1.3)	(3.22)	(2.82)	**(1.01)**	(6.56)	(1.05)	(3.10)	(4.03)	**(1.55)**
Phoneme	28.67	26.71	25.50	24.17	**21.95**	28.81	24.21	22.88	22.88	**21.95**
	(0.58)	(1.4)	(0.88)	(0.89)	**(0.71)**	(0.62)	(1.7)	(1.01)	(1.01)	**(0.68)**
Titanic	26.50	24.15	22.77	22.44	**21.69**	25.45	25.08	22.82	22.02	**21.63**
	(3.35)	(3.12)	(0.82)	(0.84)	**(0.99)**	(2.52)	(1.2)	(0.74)	(0.77)	**(1.05)**
Splice	23.57	23.48	17.25	16.12	**15.59**	23.40	22.82	16.25	16.23	**15.64**
	(1.93)	(0.76)	(0.93)	(0.97)	**(0.9)**	(2.10)	(2.63)	(1.12)	(0.97)	**(0.89)**

Table 3: Average error rate (%) and standard derivation of different aggregate losses combined with individual logistic loss and hinge loss over 5 data sets. The best results are shown in bold. (R_Max: Robust_Max)

is given in Table 2). For each data set, we first randomly select 50% samples for training, and the remaining 50% samples are randomly split for validation and testing (each contains 25% samples). Hyper-parameters C, k, and m are selected based on the validation set. Specifically, parameter C is chosen from $\{10^0, 10^1, 10^2, 10^3, 10^4, 10^5\}$, parameter $k \in \{1\} \cup [0.1 : 0.1 : 1]n$, where n is the number of training samples, and parameter m is selected in the range of $[1, k)$. The following results are based on the optimal values of k and m obtained based on the validation set. The random splitting of the training/validation/testing sets is repeated 10 times and the average error rates, as well as the standard derivation on the testing set are reported in Table 3. In (Shalev-Shwartz and Wexler, 2016), the authors introduce slack variables to indicate outliers and propose a robust version of the maximum loss. We term it as Robust_Max loss and compare it to our method as one of the baselines. As these results show, compared to the maximum, Robust_Max, average, and AT_k losses, the AoRR loss achieves the best performance on all five data sets with both individual logistic loss and individual hinge loss. For individual logistic loss, the AoRR loss significantly improves the classification performance on Monk and Phoneme data sets and a slight improvement on data sets Titanic and Splice. More specifically, the performance of maximum aggregate loss is very poor in all cases due to its high sensitivity to outliers or noisy data. The optimization of the Robust_Max loss uses convex relaxation on the domain of slack variables constraint and using l_2 norm to replace the l_1 norm in the constraint. Therefore, it can alleviate the sensitivity to outliers, but cannot exclude their influence. The average aggregate loss is more robust to noise and outliers than the maximum loss and the Robust_Max loss on all data sets. However, as data distributions may be very complicated, the average loss may sacrifice samples from rare distributions to pursue a lower loss on the whole training set and obtains sub-optimal solutions accordingly. The AT_k loss is not completely free from the influence of outliers and noisy data either, which can be observed in particular on the Monk data set. On the Monk data set, in comparison to the AT_k loss, the AoRR loss reduce the misclassification rates by 4.07% for the individual logistic loss and 3.87% for the individual hinge loss, respectively.

To further compare with the AT_k loss, we investigate the influence of m in the AoRR loss. Specifically, we select the best k value based on the AT_k results and vary m in the range of
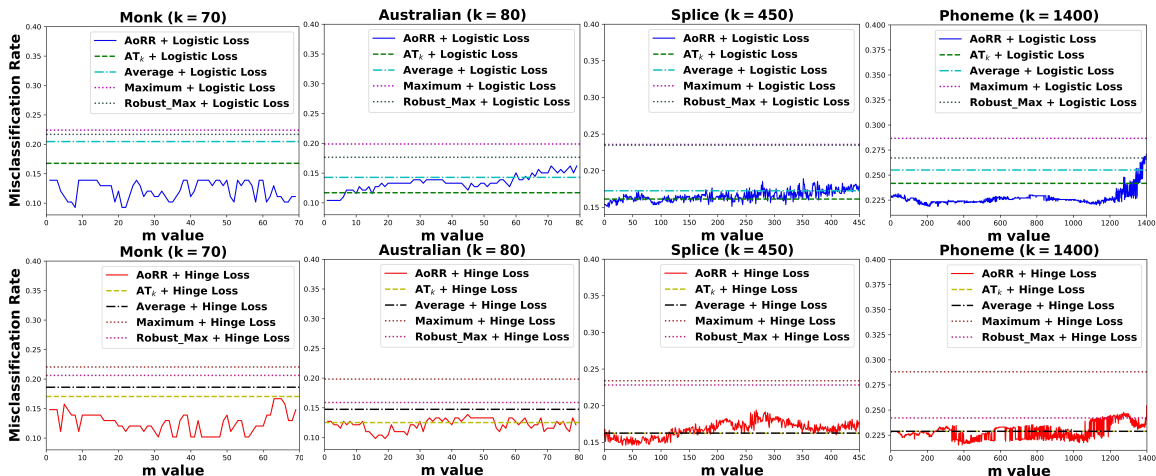
Figure 7: Tendency curves of error rate of learning AoRR loss w.r.t. m on four data sets.

$[1, k-1]$. We use the individual logistic loss and plot tendency curves of misclassification error rates w.r.t m in the first row of Figure 7, together with those from the average, maximum and Robust_Max losses. As these plots show, on all four data sets, there is a clear range of m with better performance than the corresponding AT_k loss. We observe a trend of decreasing error rates with m increasing. This is because outliers correspond to large individual losses, and excluding them from the training loss helps improve the overall performance of the learned classifier. However, when m becomes large, the classification performance is decreasing, as many samples with small losses are included in the AoRR objective and dominate the training process. Similar results based on the hinge loss can be found in the second row of Figure 7.

4.4.2 AoRR for Multi-class Classification

The AoRR aggregate loss can also be extended to multi-class classification. Let $\mathcal{Y} = \{1, \cdots, l\}$, where l represents the number of classes. We consider softmax loss as an individual loss. For sample (x_i, y_i), it is defined as

$$s_i(\theta) = -\log\Big(\frac{\exp(f_{y_i}(x_i; \theta))}{\sum_{j=1}^{l} \exp(f_j(x_i; \theta))}\Big),$$

where $f_j(x_i; \theta)$ is the prediction of j-th class, and $y_i \in \mathcal{Y}$ is the ground-truth label of x_i.

We use this multi-class classification task to verify the effectiveness of Algorithm 2. We conduct experiments on the MNIST data set (LeCun et al., 1998), which contains $60,000$ training samples and $10,000$ testing samples that are images of handwritten digits. To create a validation set, We randomly extract $10,000$ samples from training samples. Therefore, the remaining training data size is $50,000$. To simulate outliers caused by errors that occurred when labeling the data in the training set, as in the work of Wang et al. (2019), we use the symmetric (uniform) noise creation method to randomly change labels of the training samples to one of the other class labels with a given proportion. The label is chosen at random with probability $p = 0.2, 0.3, 0.4$. We use the average loss and AT_k loss as baselines. For AT_k loss, we also apply the same "warm-up" procedure and the same operation 2 to

Noise Level / Methods	0.2	0.3	0.4
Average	89.69 (0.08)	88.71 (0.10)	87.77 (0.02)
AT_k	89.71 (0.07)	88.73 (0.11)	87.62 (0.04)
AoRR	**92.42 (0.03)**	**92.26 (0.07)**	**91.87 (0.11)**

Table 4: Testing accuracy (%) of different aggregate losses combined with individual logistic loss on MNIST with different levels of symmetric noisy labels. The average accuracy and standard deviation of 5 random runs are reported and the best results are shown in bold.

Noise Level / m	0.2	0.3	0.4
Ground Truth	10000	15000	20000
Estimation	11722 (23)	16431 (17)	21026 (24)

Table 5: The estimation of hyper-parameter m. The average and standard deviation of 5 random runs are reported.

determine the hyper-parameter k. We extract the mean value plus one standard deviation of all individual losses from the validation set as operation 1's result. For operation 2, we use the mean value minus two standard deviations of all individual losses from the validation set. After multiple training epochs in the "warm-up" procedure, if the validation loss does not decrease, we stop the "warm-up" procedure. The result of this procedure can also be regarded as the result by using the average loss. After "warm-up", we do the training for AT_k loss and AoRR loss.

All models randomly run 5 times. The performance of testing accuracy is reported in Table 4. From Table 4, we can find that AoRR outperforms all other two baseline methods for all different noise levels. Specifically, our AoRR method obtains 2.71% improvement on $p = 0.2$, 3.53% improvement on $p = 0.3$, and 4.25% improvement on $p = 0.4$ when comparing to AT_k. It is obvious that Algorithm 2 works successfully and effectively. In addition, we also compare the estimated m value based on $\hat{\lambda}$ to the optimal m because we know the number of noisy data in the training set according to the noise level. From Table 5, we can find all estimated values are close to the ground truth values. It should be noted that we can modify operation 2 to slightly increase the return value $\hat{\lambda}$ so that the difference between the estimated value and the ground truth will be smaller.

5. TKML Loss for Multi-label Learning

We use SoRR to construct the individual loss for multi-label/multi-class classification, where a sample x can be associated with a set of labels $\emptyset \neq Y \subset \{1, \cdots, l\}$. Our goal is to construct a linear predictor $f_\Theta(x) = \Theta^T x$ with $\Theta = (\theta_1, \cdots, \theta_l)$. The final classifier outputs labels for x with the top k ($1 \leq k < l$) prediction scores, i.e., $\theta_{[1]}^\top x \geq \theta_{[2]}^\top x \geq \cdots \geq \theta_{[k]}^\top x$. In training, the classifier is expected to include as many true labels as possible in the top k outputs. This can be evaluated by the "margin", i.e., the difference between the $(k+1)$-th largest score of all the labels, $\theta_{[k+1]}^\top x$ and the lowest prediction score of all the ground-truth labels, $\min_{y \in Y} \theta_y^\top x$.

Data Sets	#Samples	#Features	#Labels	$\bar{c}$
Emotions	593	72	6	1.81
Scene	2,407	294	6	1.06
Yeast	2,417	103	14	4.22

Table 6: Statistical information of each data set for multi-label learning, where $\bar{c}$ represents the average number of positive labels per instance.

If we have $\theta_{[k+1]}^{\top} x < \min_{y \in Y} \theta_y^{\top} x$, then all ground-truth labels have prediction scores ranked in the top k positions. If this is not the case, then at least one ground-truth label has a prediction score not ranked in the top k. This induces the following metric for multi-label classification as $\mathbb{I}_{[\theta_{[k+1]}^{\top} x \geq \min_{y \in Y} \theta_y^{\top} x]}$. Replacing the indicator function with the hinge function and let $S(\theta) = \{s_j(\theta)\}_{j=1}^{l}$, where $s_j(\theta) = \left[1 + \theta_j^{\top} x - \min_{y \in Y} \theta_y^{\top} x\right]_{+}$, we obtain a continuous surrogate loss, as $\psi_{k,k+1}(S(\theta)) = s_{[k+1]}(\theta)$. We term this loss as the $\underline{top\text{-}k}$ $\underline{multi\text{-}label}$ (TKML) loss. According to the existing works from Crammer and Singer (2003) and Lapin et al. (2017), the conventional multi-label loss is defined as $\left[1 + \max_{y \notin Y} \theta_y^{\top} x - \min_{y \in Y} \theta_y^{\top} x\right]_{+}$. When $k = |Y|$, we have the following proposition and its proof can be found in Appendix A.6,

Proposition 6 *The* TKML *loss is a lower-bound to the conventional multi-label loss (Crammer and Singer, 2003), as* $\left[1 + \max_{y \notin Y} \theta_y^{\top} x - \min_{y \in Y} \theta_y^{\top} x\right]_{+} \geq \psi_{|Y|,|Y|+1}(S(\theta))$.

The TKML loss generalizes the conventional multi-class loss ($|Y| = k = 1$) and the top-k consistent k-guesses multi-class classification (Yang and Koyejo, 2020) ($1 = |Y| \leq k < l$). A similar learning objective is proposed in Lapin et al. (2015) corresponds to $s_{[k]}(\theta)$, however, as proved in Yang and Koyejo (2020), it is not multi-class top-k consistent. Another work in Chang et al. (2017) proposes a robust top-k multi-class SVM based on the convex surrogate of $s_{[k]}(\theta)$ to address the outliers by using a hyperparameter to cap the values of the individual losses. This approach is different from ours since we directly address the original top-k multi-class SVM problem using our TKML loss without introducing its convex surrogate and it is consistent. For a set of training data $(x_1, Y_1), \cdots, (x_n, Y_n)$, if we denote $\psi_{k,k+1}(S(x, Y; \Theta)) = \psi_{k,k+1}(S(\theta))$, the data loss on TKML can be written as $\mathcal{L}_{\text{TKML}}(\Theta) = \frac{1}{n} \sum_{i=1}^{n} \psi_{k,k+1}(S(x_i, Y_i; \Theta))$, which can be optimized using the Algorithm 1.

5.1 Experiments

We use the same ℓ_2 regularizer, $R(\theta) = \frac{1}{2C} ||\theta||_2^2$ and cross-validate hyper-parameter C in the range 10^0 to 10^5, extending it when the optimal value appears.

Multi-label classification. We use three benchmark data sets (Emotions, Scene, and Yeast) from the KEEL data repository to verify the effectiveness of our TKML loss. More details about these three data sets can be found in Table 6. For comparison, we compare TKML with logistic regression (LR) model (i.e. minimize a surrogate hamming loss (Zhang and Zhou, 2013)), and a ranking based method (LSEP, Li et al. (2017a)). For these two baseline methods, we use a sigmoid operator on the linear predictor as $f_\Theta(x) = 1/(1 + \exp(-\Theta^T x))$. Since TKML is based on the value of k, we use five different k values ($k \in \{1, 2, 3, 4, 5\}$) to evaluate the performance. For each data set, we randomly partition it to 50%/25%/25%

Data Sets	Methods	k=1	k=2	k=3	k=4	k=5
Emotions	LR	73.54(3.98)	57.48(3.35)	73.20(4.69)	86.60(3.02)	96.46(1.71)
	LSEP	72.18(4.56)	55.85(3.37)	72.18(3.74)	85.58(2.92)	95.85(1.07)
	TKML	**76.80(2.66)**	**62.11(2.85)**	**77.62(2.81)**	**90.14(2.22)**	**96.94(0.63)**
Scene	LR	73.2(0.57)	85.31(0.47)	**94.79(0.79)**	**97.88(0.63)**	**99.7(0.30)**
	LSEP	69.22(3.43)	83.83(4.83)	92.46(4.78)	96.35(3.5)	98.56(1.94)
	TKML	**74.06(0.45)**	**85.36(0.79)**	88.92(1.47)	91.94(0.87)	95.01(0.61)
Yeast	LR	**77.57(0.91)**	**70.59(1.16)**	**52.65(1.23)**	43.26(1.16)	43.49(1.33)
	LSEP	75.5(1.03)	66.84(2.9)	49.72(1.26)	41.90(1.91)	43.01(1.02)
	TKML	76.94(0.49)	67.19(2.79)	45.41(0.71)	**43.47(1.06)**	**44.69(1.14)**

Table 7: Top k multi-label accuracy with its standard derivation (%) on three data sets. The best performance is shown in bold.

Data Sets Methods	Emotions	Scene	Yeast
LR	74.85	71.6	73.56
LSEP	82.66	85.43	74.26
TKML	**84.82**	**86.38**	**74.32**

Table 8: AP (%) results on three data sets. The best performance is shown in bold.

samples for training/validation/testing, respectively. This random partition is repeated 10 times, and the average performance on testing data is reported in Table 7. We use a metric (top k multi-label accuracy) $\frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{[(Z_i \subseteq Y_i) \vee (Y_i \subseteq Z_i)]}$ to evaluate the performance, where n is the size of the sample set. For instance, (x_i, Y_i) with Y_i be its ground-truth set, $f_\Theta(x_i) \in \mathbb{R}^l$ be its predicted scores, and Z_i be a set of top k predictions according to $f_\Theta(x_i)$. This metric reflects the performance of a classifier can get as many true labels as possible in the top k range. More settings can be found in Appendix B.3.

From Table 7, we note that the TKML loss in general improves the performance on the Emotions data set for all different k values. These results illustrate the effectiveness of the TKML loss. More specifically, our TKML method obtains 4.63% improvement on $k = 2$ and 4.42% improvement on $k = 3$ when comparing to LR. This rate of improvement becomes higher (6.26% improvement on $k = 2$) when compare to LSEP. We also compare the performance of the method based on the TKML loss on different k values. If we choose the value of k close to the number of the ground-truth labels, the corresponding classification method outperforms the two baseline methods. For example, in the case of the Emotions data set, the average number of positive labels per instance is 1.81, and our method based on the TKML loss achieves the best performance for $k = 1, 2$. As another example, the average number of true labels for the Yeast data set is 4.22, so the method based on the TKML loss achieves the best performance for $k = 4, 5$.

We also adopt a widely used multi-label learning metric named average precision (AP) for performance evaluation. It is calculated by (Zhang and Zhou, 2013)

$$\text{AP} = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{|Y_i|} \sum_{j \in Y_i} \frac{|\{\tau \in Y_i | rank_f(x_i, \tau) < rank_f(x_i, j)\}|}{rank_f(x_i, j)},$$

where $rank_f(x_i, j)$ returns the rank of $f_j(x_i)$ in descending according to $\{f_a(x_i)\}_{a=1}^l$. From Table 8, we can find our TKML method outperforms the other two baseline approaches on all

Noise Level	Methods	Top-1 Acc.	Top-2 Acc.	Top-3 Acc.	Top-4 Acc.	Top-5 Acc.
0.2	SVM_α	78.33(0.18)	90.66(0.29)	95.12(0.2)	97.28(0.09)	98.49(0.1)
	TKML	**83.06(0.94)**	**94.17(0.19)**	**97.24(0.13)**	**98.47(0.05)**	**99.22(0.01)**
0.3	SVM_α	74.65(0.17)	89.31(0.24)	94.14(0.2)	96.73(0.23)	98.19(0.07)
	TKML	**80.13(1.24)**	**93.37(0.1)**	**96.81(0.22)**	**98.21(0.05)**	**99.08(0.05)**
0.4	SVM_α	68.32(0.32)	86.71(0.42)	93.14(0.49)	96.16(0.32)	97.84(0.18)
	TKML	**75(1.15)**	**92.41(0.14)**	**96.2(0.13)**	**97.95(0.1)**	**98.89(0.04)**

Table 9: Testing accuracy (%) of two methods on MNIST with different levels of asymmetric noisy labels. The average accuracy and standard deviation of 5 random runs are reported and the best results are shown in bold. (Acc.: Accuracy)
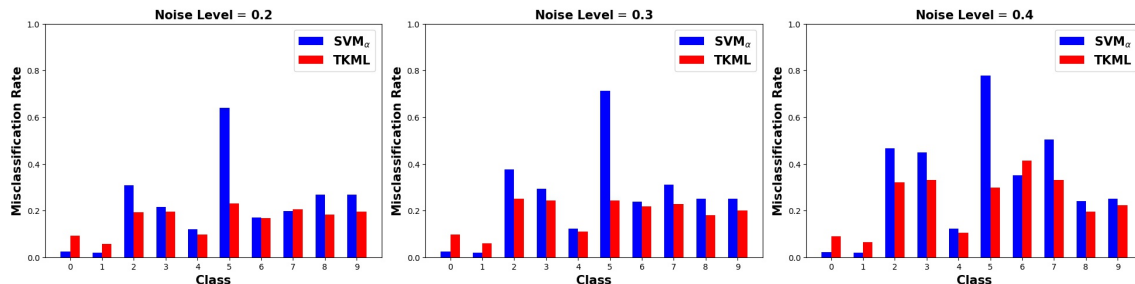

Figure 8: The class-wise error rates of two methods with different noise level data.

data sets. For the Emotions data set, the AP score of TKML is 2.16% higher than the LSEP method and near 10% higher than the LR. The performance is also slightly improved on Scene and Yeast data sets. These results demonstrate the effectiveness of our TKML method.

Robustness analysis. As a special case of AoRR, the TKML loss exhibit similar robustness with regards to outliers, which can be elucidated with experiments in the multi-class setting (*i.e.*, k=1 and $|Y|$=1). We conduct experiments on the MNIST data set (LeCun et al., 1998), which contains $60,000$ training samples and $10,000$ testing samples. To simulate outliers caused by errors that occurred when labeling the data, as in the work of Wang et al. (2019), we use the asymmetric (class-dependent) noise creation method (Patrini et al., 2017; Zhang and Sabuncu, 2018) to randomly change labels of the training data ($2\rightarrow7$, $3\rightarrow8$, $5\leftrightarrow6$, and $7\rightarrow1$) with a given proportion. The flipping label is chosen at random with probability $p = 0.2, 0.3, 0.4$. As a baseline, we use the top-k multi-class SVM (SVM_α) (Lapin et al., 2015). The performance is evaluated with the top 1, top 2, $\cdots$, top 5 accuracy on the testing samples. More details about the settings can be found in Appendix B.4.

From Table 9, it is clear that our method TKML consistently outperforms the baseline SVM_α among all top 1-5 accuracies. The gained improvement in performance is getting more significant as the level of noise increases. Since our flipping method only works between two different labels, we expected the performance of TKML has some significant improvements on top 1 and 2 accuracies. Indeed, this expectation is correctly verified as Table 9 clearly indicates that the performance of our method is better than SVM_α by nearly 7% accuracy (see Top-1 accuracy in the noise level 0.4). These results also demonstrate our optimization framework works well.

To evaluate our method is better than SVM_α especially on the flipped class, we plot the class-wise error rate w.r.t different noise level data. As seen in Figure 8, our method TKML outperforms SVM_α on the flipping classes such as 2 and 3, especially in class 5. As the noise level increases, the performance gap becomes more pronounced. For flipping class 7, the performance in this class is increased when the noise level increases from 0.3 to 0.4. TKML also gets good performance in class 6 when the noise level is 0.2 and 0.3.

6. Combination of AoRR and TKML

The TKML individual loss applies to the multi-label learning problem at the label level. However, it may also be the case that the training samples for a multi-label learning problem include outliers. It is thus a natural idea to combine the TKML individual loss at the label level and the AoRR aggregate loss at the sample level to construct a more robust learning objective.

To the best of our knowledge, considering robustness for multi-label learning at both the label and the sample level has not been extensively studied in the literature. The most relevant work is Rawat et al. (2020), which considers robustness at both levels for multi-class classification (*i.e.*, out of multiple class labels, only one is the true label). The doubly-stochastic mining method in Rawat et al. (2020) uses the AT_k aggregate loss on the sample level and top-k multi-class SVM loss on the label level. As we discussed before, AT_k loss cannot eliminate the outliers and top-k multi-class SVM does not satisfy the multi-class top-k consistent. Furthermore, the proposed method from their work cannot deal with the top-k multi-label learning problem where there are multiple true labels for each instance.

We consider a learning objective that combines AoRR and TKML for multi-label learning. Theorem 2 tells us that we can get the same optimal solution of SoRR problem (equation (1)) by solving the problem of equation (4). However, it is hard to deal with the cardinality constraint $\|q\|_0 = n - m$ in equation (4). Thus, we need to reformulate the equation (4) in order to remove the cardinality constraint. Before we introduce the new formulation, we provide an useful Lemma as follows,

Lemma 7 $\sum_{i=m+1}^{n} s_{[i]}$ *is a concave function of the elements of S. Furthermore, we have* $\sum_{i=m+1}^{n} s_{[i]} = \max_{\lambda \in \mathbb{R}} \left\{ (n-m)\lambda - \sum_{i=1}^{n} [\lambda - s_i]_+ \right\}$, *of which* $s_{[m]}$ *is an optimum solution.*

Proof can be found in Appendix A.7. According to equation (4), we can substitute the optimal q^* to the constraint. Using Lemma 7, we get

$$\min_\lambda (k - m)\lambda + \sum_{i=m+1}^{n} [[s(\theta) - \lambda]_+]_{[i]}$$

$$= \min_\lambda (k - m)\lambda + \max_{\hat{\lambda}} \left\{ (n - m)\hat{\lambda} - \sum_{i=1}^{n} [\hat{\lambda} - [s_i(\theta) - \lambda]_+]_+ \right\}.$$

Algorithm 3: Combination of `AoRR` and `TKML`

---

1 **Initialization:** $\theta^{(0)}$, $\lambda^{(0)}$, $\hat{\lambda}^{(0)}$, η_t, k', k, and m

2 **for** $t = 0, 1, ...$ **do**

3   Compute $s_i(\theta^{(t)}) = [1 + \theta^{(t)\top}_{[k'+1]} x_i - \min_{y \in Y_i} \theta^{(t)\top}_y x_i]_+$, $\forall i \in \{1, \cdots, n\}$

4   Update parameters $\theta^{(t+1)}$, $\lambda^{(t+1)}$, and $\hat{\lambda}^{(t+1)}$ with equation (12)

5 **end**

Finally, we obtain a new objective function that the combination of `TKML` and `AoRR` as follows,

$$\mathcal{L}_{\text{TKML}-\text{AoRR}}(S(\theta)) = \frac{\psi_{m,k}(S(\theta))}{k-m} = \min_\lambda \max_{\hat{\lambda}} \ \lambda + \frac{n-m}{k-m}\hat{\lambda} - \frac{1}{k-m}\sum_{i=1}^n [\hat{\lambda} - [s_i(\theta) - \lambda]_+]_+$$

$$= \frac{1}{k-m} \min_\lambda \max_{\hat{\lambda}} \ (k-m)\lambda - m\hat{\lambda} + \sum_{i=1}^n \hat{\lambda} - [\hat{\lambda} - [s_i(\theta) - \lambda]_+]_+.$$

$$(11)$$

For $s_i(\theta)$, we use `TKML` individual loss to replace it. So we have $s_i(\theta) = [1 + \theta^\top_{[k'+1]} x_i - \min_{y \in Y_i} \theta^\top_y x_i]_+$, where we use k' in the label level to distinguish k in the sample level. Then, we develop a heuristic algorithm for learning $\mathcal{L}_{\text{TKML}-\text{AoRR}}(S(\theta))$. For a set of training data $\{(x_i, Y_i)\}_{i=1}^n$, where sample x_i is associated with a set of labels $\emptyset \neq Y_i \subset \{1, \cdots, l\}$. The algorithm iteratively updates the parameters θ, λ and $\hat{\lambda}$ based on the $\mathcal{L}_{\text{TKML}-\text{AoRR}}(S(\theta))$ over training samples with the following steps:

$$\theta^{(t+1)} = \theta^{(t)} - \eta_l \left(\frac{1}{k-m} \sum_{i=1}^n \partial s_i(\theta^{(t)}) \cdot \mathbb{I}_{[\hat{\lambda}^{(t)} > [s_i(\theta^{(t)}) - \lambda^{(t)}]_+]} \cdot \mathbb{I}_{[s_i(\theta^{(t)}) > \lambda^{(t)}]} \right),$$

$$\lambda^{(t+1)} = \lambda^{(t)} - \eta_l \left(1 - \frac{1}{k-m} \sum_{i=1}^n \mathbb{I}_{[\hat{\lambda}^{(t)} > [s_i(\theta^{(t)}) - \lambda^{(t)}]_+]} \cdot \mathbb{I}_{[s_i(\theta^{(t)}) > \lambda^{(t)}]} \right), \qquad (12)$$

$$\hat{\lambda}^{(t+1)} = \hat{\lambda}^{(t)} + \eta_l \left(\frac{n-m}{k-m} - \frac{1}{k-m} \sum_{i=1}^n \mathbb{I}_{[\hat{\lambda}^{(t)} > [s_i(\theta^{(t)}) - \lambda^{(t)}]_+]} \right),$$

where $0 \leq m < k \leq n$, $1 \leq k' < l$, η_l is the step size, and $\partial s_i(\theta^{(t)})$ denotes the (sub)gradient of $s_i(\theta^{(t)})$ with respect to $\theta^{(t)}$. Algorithm 3 describes the pseudo-code of the proposed heuristic algorithm. For a given loop size $|t|$ and training sample size n, the time complexity of Algorithm 3 is $O(|t| \cdot n)$.

It should be mentioned that the mini-batch SGD method can be applied to Algorithm 3 when the training sample size is large. We can also use more complex non-linear models such as deep neural networks to replace the linear model in the `TKML` individual loss $s_i(\theta)$. Furthermore, Algorithm 3 with mini-batch B setting will become a general algorithm. It can be generalized to other types of existing algorithms for multi-class learning with setting different hyper-parameter values of k', k, and m. For example, if $|Y_i| = 1 = k', \forall i$, and $m = 0$, it becomes OSGD algorithm (Kawaguchi and Lu, 2020), which updates the model parameter based on the average loss over the top-k largest individual losses in the mini-batch.

Noise Level	Methods	$k'=1$	$k'=2$	$k'=3$	$k'=4$	$k'=5$
0	TKML-Average	73.78 (1.59)	73.64 (1.53)	43.90 (2.99)	34.94 (3.78)	43.55 (0.96)
	TKML-AT$_k$	74.38 (1.56)	73.84 (1.25)	49.44 (1.76)	43.16 (2.16)	45.99 (2.13)
	TKML-AoRR	**74.94 (1.82)**	**73.88 (1.26)**	**50.25 (1.39)**	**45.93 (2.15)**	**46.13 (1.77)**
0.1	TKML-Average	73.08 (1.55)	73.24 (1.52)	43.80 (4.78)	34.22 (4.72)	43.48 (0.65)
	TKML-AT$_k$	74.28 (1.36)	73.44 (1.34)	49.26 (0.80)	41.12 (1.91)	45.72 (1.97)
	TKML-AoRR	**74.55 (1.62)**	**73.68 (1.52)**	**50.19 (0.81)**	**45.52 (2.35)**	**46.05 (2.20)**
0.2	TKML-Average	72.71 (3.68)	72.64 (0.88)	43.38 (2.00)	33.96 (4.31)	43.30 (1.09)
	TKML-AT$_k$	73.88 (1.03)	73.24 (1.25)	49.15 (1.45)	39.24 (1.15)	45.62 (1.70)
	TKML-AoRR	**74.36 (1.61)**	**73.50 (1.38)**	**49.73 (1.41)**	**45.48 (2.35)**	**46.00 (1.59)**
0.3	TKML-Average	71.32 (3.86)	71.94 (1.06)	43.16 (2.78)	33.78 (3.03)	41.44 (4.09)
	TKML-AT$_k$	73.78 (1.09)	73.04 (1.21)	47.48 (1.90)	36.90 (1.91)	45.06 (2.01)
	TKML-AoRR	**74.31 (1.40)**	**73.48 (1.36)**	**49.69 (1.07)**	**44.71 (2.11)**	**45.79 (1.98)**

Table 10: Top k multi-label accuracy and its standard derivation (%) on the Yeast data set with different levels of symmetric noisy labels. The average best performance is shown in bold based on 10 random runs.

If $|Y_i| = 1 = k', \forall i$, $m = |B| - k$, and $k = |B|$, it becomes ITLM algorithm (Shen and Sanghavi, 2019), which selects the bottom-k smallest individual losses in each mini-batch and then uses them to update the model parameter. If $|Y_i| = 1 = k', \forall i$, $m = |B| - 1$, and $k = |B|$, our algorithm is MKL-SGD (Shah et al., 2020). This method selects a sample with the smallest individual loss in the mini-batch set, then use the gradient of this individual loss to update the model parameters. In addition, if $|Y_i| = 1, \forall i$, and $m = 0$, our Algorithm is similar to the doubly-stochastic mining method in Rawat et al. (2020).

6.1 Experiments

The Yeast data set is selected to show the efficiency of our loss function equation (11) and Algorithm 3. We randomly split Yeast data into two parts, which are 80% samples for training and 20% samples for testing. This random partition is repeated 10 times. The average performances on testing data are reported and they are based on the optimal values of hyper-parameters. To compare the combined TKML and AoRR method (TKML-AoRR), we consider other two combination methods, which use the average (TKML-Average) and the AT$_k$ (TKML-AT$_k$) aggregate loss on the sample level for top-k multi-label learning. Assuming the training data size is n, the hyper-parameter k is selected in the range of $[1, n)$ for TKML-AT$_k$ method. After finding the optimal value k for TKML-AT$_k$ method, we apply it to our TKML-AoRR method. Then hyper-parameter m in our method is selected in the range of $[1, k)$. Note that we can also apply the method from Section 4.3 to determine the hyper-parameters k and m in practice. However, we use this grid search method to show the performance changes of our method as the hyper-parameters change in the experiment. On the other hand, since we use AoRR aggregate method, we should expect that our method also exhibits robustness with regards to outliers. Therefore, we introduce noise data to the training set of Yeast. Following Li et al. (2020b), we generate symmetric label noise. Specifically, we randomly choose training samples with probability $p = 0.1, 0.2, 0.3$ and change each of their labels to another random label. We use top-k multi-label accuracy as a metric to evaluate the performance as we have applied in Section 5.1. All results are shown in the Table 10. More details about the settings can be found in Appendix B.5.
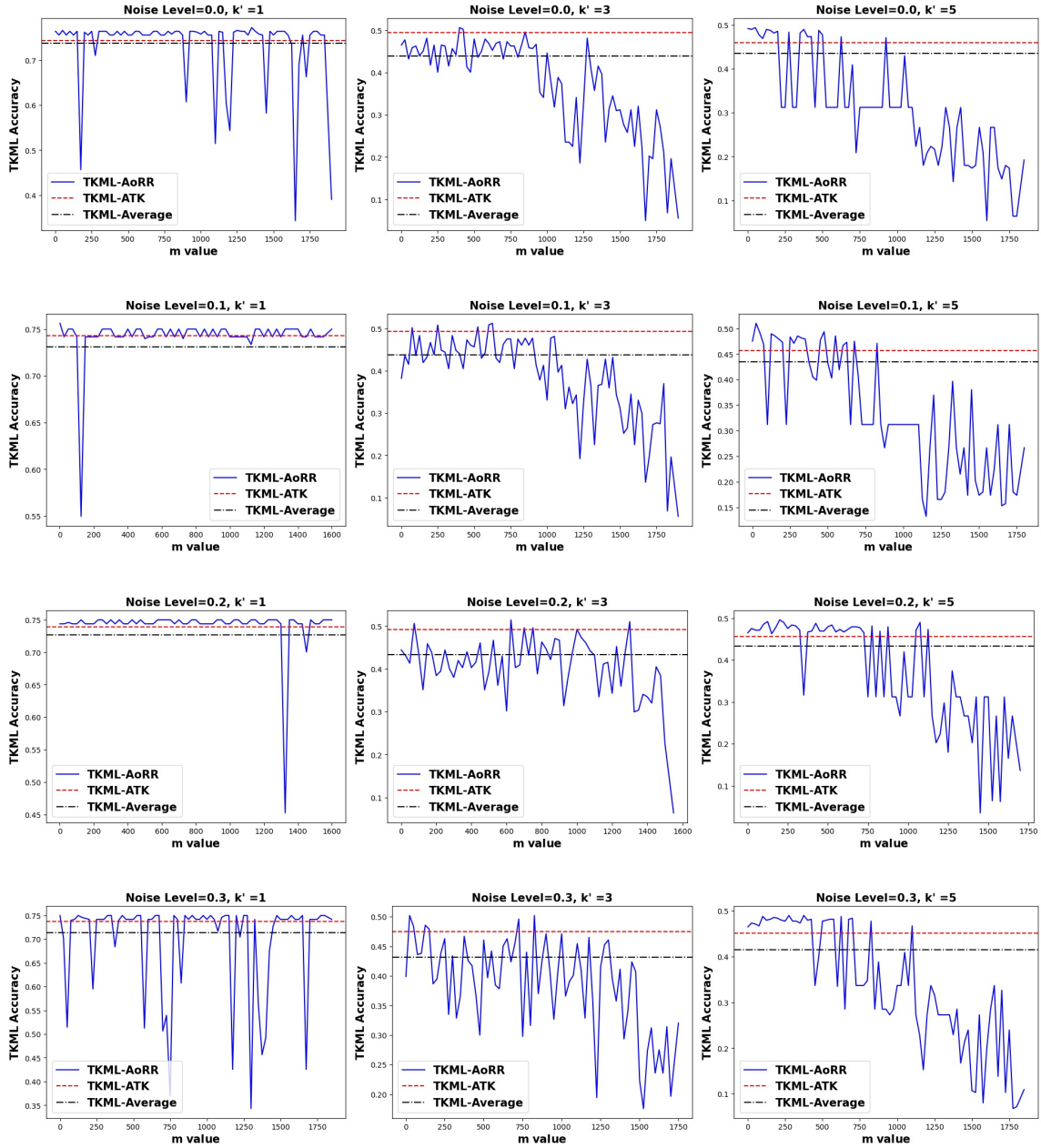
Figure 9: Tendency curves of `TKML` accuracy of learning `TKML-AoRR` loss w.r.t. m on four noise level settings ($p = 0.0, 0.1, 0.2, 0.3$). The left figures are based on the $k' = 1$ setting. The middle figures are based on the $k' = 3$ setting. The right figures are based on the $k' = 5$ setting.

From Table 10, we can find that our `TKML-AoRR` method outperforms the other two methods in all noise levels and all k' values. Especially, there is more than 10% improvement when compared with `TKML-Average` method in $k' = 4$. The results show that our algorithm works efficiently. As the noise level increases, the top-k multi-label accuracy of the same

29

method is decreasing. The performance declines slowly with the increase of noise level while using our method. For example, from noise level 0 to noise level 0.3, the performance only has less than 1% difference in $k' = 1, 2, 3, 5$. For $k' = 4$, the gap is 1.22% from noise level 0 to 0.3 using our method. However, the performance is decreased near 10% for the TKML-AT$_k$ method. As we mentioned before, the performance of the TKML-AT$_k$ method is better than the TKML-Average method. But it cannot eliminate the influence of the outliers. Therefore, the performance can be further improved by using our method. To see how the performance change with setting different values of the hyper-parameter m, we show the tendency curves of TKML accuracy for the learning TKML-AoRR loss w.r.t. m in Figure 9. From Figure 9, we can find there is a clear range of m with better performance than the corresponding TKML-Average and TKML-AT$_k$ loss.

7. Conclusion

In this work, we have introduced a general approach to form learning objectives, *i.e.*, the sum of ranked range, which corresponds to the sum of a consecutive sequence of sorted values of a set of real numbers. We show that SoRR can be expressed as the difference between two convex problems and optimized with the difference-of-convex algorithm (DCA). We also show that SoRR can be reformed as a bilevel optimization problem.

We explored two applications in machine learning of the minimization of the SoRR framework, namely the AoRR aggregate loss for binary/multi-class classification at the data sample level and the TKML individual loss for multi-label/multi-class classification at the data label level. For the AoRR aggregate loss, we discussed its connection with CVaRs and explore its generalization bound. We also proposed a method for learning hyper-parameters, which can make AoRR works more efficiently in practice. Furthermore, we combine the AoRR aggregate loss at the sample level and the TKML individual loss at the label level to enhance the robustness in the top-k multi-label learning. A heuristic algorithm is proposed to optimize the combined loss. We conducted extensive experiments to show the effectiveness of the proposed frameworks on achieving superior generalization and robust performance on synthetic and real data sets.

There are several potential **limitations** to our proposed methods. First, Algorithm 1 and Algorithm 2 are based on the DCA. We use a stochastic sub-gradient method to iterative optimize the model parameter in the inner loop, which may introduce bias about the optimal value of the learned model parameters θ. Furthermore, using the inner loop to find the optimal solution in each outer loop may introduce a high time complexity. As we analyzed the time complexity of Algorithm 1 and Algorithm 2, both of them are increased by increasing the size of the inner loop size $|l|$. Second, in Algorithm 2, we need a clean validation set to learn the hyper-parameters k and m. However, it is very hard to guarantee the extracted validation set from the training set is clean. If it is not clean, the learned result through Algorithm 2 may not reliable. Third, although experiments show that Algorithm 3 is successful and effective, its theoretical convergence has not been studied. The learning objective (11) is a minimax optimization problem, which is worth exploring its convergence property and optimization error.

For **future works**, we will try to address the above-mentioned limitations of our proposed methods. We also plan to further study the statistical consistency of TKML loss for multi-label

learning and the extension of our `SoRR` framework with deep neural network structures. Testing the vulnerability of the deep learning models that incorporate the `TKML` loss is also worthy to explore. We have done such works in Hu et al. (2021). However, how to protect `TKML` based models against adversarial attacks is important for future work.

Acknowledgments

Appendix A. Proofs

A.1 Proof of Theorem 1

Proof To prove Theorem 1, we need the following lemma.

Lemma A.1 (Ogryczak and Tamir (2003)) . $\phi_k(S)$ *is a convex function of the elements of S. Furthermore, for any $i \in [1,n]$, we have $\sum_{i=1}^{k} s_{[i]} = min_{\lambda \in \mathbb{R}} \{k\lambda + \sum_{i=1}^{n} [s_i - \lambda]_+\}$, of which $s_{[k]}$ is an optimum solution.*

From Lemma A.1, we have

$$\min_{\theta} \psi_{m,k}(S(\theta)) = \min_{\theta} \left[\phi_k(S(\theta)) - \phi_m(S(\theta)) \right]$$

$$= \min_{\theta} \left[\min_{\lambda \in \mathbb{R}} \left\{ k\lambda + \sum_{i=1}^{n} [s_i(\theta) - \lambda]_+ \right\} - \min_{\hat{\lambda} \in \mathbb{R}} \left\{ m\hat{\lambda} + \sum_{i=1}^{n} [s_i(\theta) - \hat{\lambda}]_+ \right\} \right].$$

If the optimal solution θ^* is achieved, from Lemma A.1, we get $\lambda = s_{[k]}$ and $\hat{\lambda} = s_{[m]}$. Therefore, $\hat{\lambda} > \lambda$ because $k > m$. ∎

A.2 Proof of Equation (3)

Proof Before introducing the sub-gradient of $\phi_m(S(\theta))$, we provide a very useful characterization of differentiable properties of the optimal value function (Bertsekas, 1971, Proposition A.22), which is also an extension of Danskin's theorem (Danskin, 2012).

Lemma A.2 *Let $\phi : \mathbb{R}^n \times \mathbb{R}^m \to (-\infty, \infty]$ be a function and let Y be a compact subset of $\mathbb{R}^m$. Assume further that for every vector $y \in Y$ the function $\phi(\cdot, y) : \mathbb{R}^n \to (-\infty, \infty]$ is a closed proper convex function. Consider the function f defined as $f(x) = sup_{y \in Y} \phi(x, y)$, then if f is finite somewhere, it is a closed proper convex function. Furthermore, if $int(domf) \neq \emptyset$ and ϕ is continuous on the set $int(domf) \times Y$, then for every $x \in int(domf)$ we have $\partial f(x) = conv\{\partial \phi(x, \overline{y}) | \overline{y} \in \overline{Y}(x)\}$, where $\overline{Y}(x)$ is the set $\overline{Y}(x) = \{\overline{y} \in Y | \phi(x, \overline{y}) = max_{y \in Y} \phi(x, y)\}$*

We apply Lemma A.2 with a new notation $\phi_m(\theta, \hat{\lambda}) = m\hat{\lambda} + \sum_{i=1}^{n} [s_i(\theta) - \hat{\lambda}]_+$. Suppose $\theta \in \mathbb{R}^n$ and $\hat{\lambda} \in \mathbb{R}$, the function $\phi_m : \mathbb{R}^n \times \mathbb{R} \to (-\infty, \infty]$. Let Y be a compact subset of $\mathbb{R}$ and for every $\hat{\lambda} \in Y$, it is obvious that the function $\phi_m(\cdot, \hat{\lambda}) : \mathbb{R}^n \to (-\infty, \infty]$ is a closed proper convex function w.r.t θ from the second term of Eq.(1).

Consider a function f defined as $f(\theta) = sup_{\hat{\lambda} \in Y} \phi(\theta, \lambda)$, since f is finite somewhere, it is a closed proper convex function. The interior of the effective domain of f is nonempty, and that ϕ_m is continuous on the set $int(domf) \times Y$. The condition of lemma A.2 is satisfied.

$\forall \theta \in int(domf)$, we have

$$\partial f(\theta) = conv\{\partial \phi_m(\theta, \overline{\lambda}) | \overline{\lambda} \in \overline{Y}(\theta)\},$$

where

$$\overline{Y}(\theta) = \{\overline{\lambda} \in Y | \phi_m(\theta, \overline{\lambda}) = max_{\hat{\lambda} \in Y} \phi_m(\theta, \hat{\lambda})\} = \{\overline{\lambda} \in Y | -\phi_m(\theta, \overline{\lambda}) = -min_{\hat{\lambda} \in Y} \phi_m(\theta, \hat{\lambda})\}.$$

As we know $-min_{\hat{\lambda} \in Y} \phi_m(\theta, \hat{\lambda}) = -\phi_m(S(\theta))$. This means the subdifferential of f w.r.t θ exists when we set the optimal value of $\hat{\lambda}$.

From the above and the lemma A.1, we can get the sub-gradient $\hat{\theta} \in \partial \phi_m(S(\theta)) = \sum_{i=1}^{n} \partial s_i(\theta) \cdot \mathbb{I}_{[s_i(\theta) > \hat{\lambda}]}$, where $\hat{\lambda}$ equals to $s_{[m]}(\theta)$. ∎

A.3 Proof of Theorem 2

Proof The proof of this proposition is similar to the proof of proposition 1 in the paper Fujiwara et al. (2017). However, they only discuss the linear case of $s_i(\theta)$. Under the constraints, we can rewrite the formula in Theorem 2 as

$$(k-m)\lambda + \sum_{i=1}^{n} q_i[s_i(\theta) - \lambda]_+ = (k-m)\lambda + \sum_{i=1}^{n}[s_i(\theta) - \lambda]_+ - \sum_{i=1}^{n}(1 - q_i)[s_i(\theta) - \lambda]_+$$

$$= k\lambda + \sum_{i=1}^{n}[s_i(\theta) - \lambda]_+ - \sum_{i=1}^{n}(1 - q_i)\{[s_i(\theta) - \lambda]_+ + \lambda\}.$$

The last equality holds because $\sum_{i=1}^{n}(1 - q_i) = n - (n - m) = m$.

For the term $\sum_{i=1}^{n}(1-q_i)\{[s_i(\theta)-\lambda]_+ + \lambda\}$, we assume $s_i(\theta^*), \forall i$, are sorted in descending order when getting the optimal model parameter θ^*. For example, $s_1(\theta^*) \geq s_2(\theta^*) \geq \cdots \geq s_n(\theta^*)$. Since $\lambda^* \geq 0$, the optimal q^* should be $q_1^* = \cdots = q_m^* = 0$, $q_{m+1}^* = \cdots = q_n^* = 1$. Note that λ^* must be an optimal solution of the problem

$$\min_{\lambda}(k - m)\lambda + \sum_{i=m+1}^{n} q_i^*[s_i(\theta^*) - \lambda]_+.$$

From Lemma A.1, we know $s_{m+1}(\theta^*) \geq \lambda^*$, which implies that $s_i(\theta^*) - \lambda^* \geq 0$ holds for $q_i < 1$. Therefore, $\sum_{i=1}^{n}(1 - q_i)\{[s_i(\theta) - \lambda]_+ + \lambda\} = \sum_{i=1}^{n}(1 - q_i)s_i(\theta)$. Furthermore, we know

$$\min_{\hat{\lambda}} \left\{ m\hat{\lambda} + \sum_{i=1}^{n}[s_i(\theta) - \hat{\lambda}]_+ \right\} = \max_{q} \left\{ \sum_{i=1}^{n}(1 - q_i)s_i(\theta) \,\middle|\, q_i \in [0, 1], \, ||q||_0 = n - m \right\}.$$

Then we get

$$\min_{\lambda,q}(k - m)\lambda + \sum_{i=1}^{n} q_i[s_i(\theta) - \lambda]_+ \quad s.t. \, q_i \in [0, 1], \, ||q||_0 = n - m$$

$$= \min_{\lambda} \left\{ k\lambda + \sum_{i=1}^{n}[s_i(\theta) - \lambda]_+ \right\} - \min_{\hat{\lambda}} \left\{ m\hat{\lambda} + \sum_{i=1}^{n}[s_i(\theta) - \hat{\lambda}]_+ \right\}.$$

∎

A.4 Proof of Theorem 4

Proof Without loss of generality, by normalization we can assume $s(0) = 1$ which can be satisfied by scaling. For any fixed $x \in \mathcal{X}$, by the definition of $f^* = \arg\inf \mathcal{L}(f, \lambda^*, \hat{\lambda}^*)$, we know that

$$f^*(x) = t^* = \arg\inf_{t \in \mathbb{R}} \mathbb{E}\Big[[s(Yt) - \lambda^*]_+ - [s(Yt) - \hat{\lambda}^*]_+ \Big| X = x \Big].$$

Notice the assumption $\hat{\lambda}^* > \lambda^*$ and recall $\eta(x) = P(y = 1|x)$. We need to show that $t^* > 0$ for $\eta(x) > 1/2$ and $t^* < 0$ if $\eta(x) < 1/2$. Indeed, if $t^* \neq 0$, then, by the definition of t^*, we have that

$$\mathbb{E}\Big[[s(Yt^*) - \lambda^*]_+ - [s(Yt^*) - \hat{\lambda}^*]_+ \Big| X = x \Big] < \mathbb{E}\Big[[s(-Yt^*) - \lambda^*]_+ - [s(-Yt^*) - \hat{\lambda}^*]_+ \Big| X = x \Big].$$

The above inequality is identical to

$$\Big[\big((s(t^*) - \lambda^*)_+ - (s(t^*) - \hat{\lambda}^*)_+ \big) - \big((s(-t^*) - \lambda^*)_+ - (s(-t^*) - \hat{\lambda}^*)_+ \big) \Big] [2\eta(x) - 1] < 0.$$

Since $\hat{\lambda}^* > \lambda^*$, we have that that $g(s) = (s - \lambda^*)_+ - (s - \hat{\lambda}^*)_+$ is a non-decreasing function of variable s. Then, if $\eta(x) > \frac{1}{2}$ we must have $g(s(t^*)) < g(s(-t^*))$ which indicates $s(t^*) < s(-t^*)$. From the non-increasing property of s on $\mathbb{R}$, $s(t)$ is also a convex function and $s'(0) < 0$ immediately indicates $t^* > 0$. Likewise, we can show that $t^* < 0$ for $\eta(x) < 1/2$.

To prove $t = 0$ is not a minimizer, without loss of generality, assume $\eta(x) > \frac{1}{2}$. We need to consider two conditions as follows,

1. If $0 \leq \lambda^* < \hat{\lambda}^* \leq 1$ and $s(0) = 1$, then

$$\begin{aligned}
A &= \mathbb{E}\Big[[s(0) - \lambda^*]_+ - [s(0) - \hat{\lambda}^*]_+ \Big| X = x \Big] \\
&= [1 - \lambda^*]_+ - [1 - \hat{\lambda}^*]_+ \\
&= \hat{\lambda}^* - \lambda^*.
\end{aligned}$$

Since $s'(0) < 0$ and s is non-increasing, there exists $t^0 > t^* = 0 > -t^0$, and $s(-t^0) > s(0) \geq \hat{\lambda}^* > s(t^0) > \lambda^*$. Let

$$\begin{aligned}
B &= \mathbb{E}\Big[[s(Yt^0) - \lambda^*]_+ - [s(Yt^0) - \hat{\lambda}^*]_+ \Big| X = x \Big] \\
&= \Big([s(t^0) - \lambda^*]_+ - [s(t^0) - \hat{\lambda}^*]_+ \Big) \eta(x) + \Big([s(-t^0) - \lambda^*]_+ - [s(-t^0) - \hat{\lambda}^*]_+ \Big) \big(1 - \eta(x) \big) \\
&= \Big([s(-t^0) - \lambda^*]_+ - [s(-t^0) - \hat{\lambda}^*]_+ \Big) \\
&\quad + \Big[\big([s(t^0) - \lambda^*]_+ - [s(t^0) - \hat{\lambda}^*]_+ \big) - \big([s(-t^0) - \lambda^*]_+ - [s(-t^0) - \hat{\lambda}^*]_+ \big) \Big] \eta(x) \\
&= \hat{\lambda}^* - \lambda^* + \Big[s(t^0) - \lambda^* - (\hat{\lambda}^* - \lambda^*) \Big] \eta(x).
\end{aligned}$$

Then

$$B - A = (s(t^0) - \hat{\lambda}^*)\eta(x) < 0.$$

Therefore, $t = 0$ is not a minimizer.

2. If $0 \leq \lambda^* \leq 1 < \hat{\lambda}^*$ and $s(0) = 1$, then

$$\frac{d}{dt}\mathbb{E}[[s(Yt) - \lambda^*]_+ - [s(Yt) - \hat{\lambda}^*]_+]|_{t=0}$$

$$= \frac{d}{dt}[\eta(x)([s(t) - \lambda^*]_+ - [s(t) - \hat{\lambda}^*]_+) + (1 - \eta(x))([s(-t) - \lambda^*]_+ - [s(-t) - \hat{\lambda}^*]_+)]|_{t=0}$$

$$= \frac{d}{dt}[\eta(x)(s(t) - \lambda^*) + (1 - \eta(x))(s(-t) - \lambda^*)]|_{t=0}$$

$$= [\eta(x)s'(t) - (1 - \eta(x))s'(-t)]|_{t=0}$$

$$= (2\eta(x) - 1)s'(0) < 0.$$

Thus $t = 0$ is not a minimizer. $\blacksquare$

A.5 Proof of Theorem 5

Proof First, we introduce a lemma from Brown (2007) and (Thomas and Learned-Miller, 2019, Theorem 1 and 2) as follows,

Lemma A.3 (*Brown (2007); Thomas and Learned-Miller (2019)*) *Let $C_\alpha[s] = \mathbb{E}[s|s \geq V_\alpha[s]]$ and $\widehat{C}_\alpha[s] := \inf_{\lambda \in \mathbb{R}}\{\lambda + \frac{1}{n\alpha}\sum_{i=1}^n [s_i - \lambda]_+\}$, if $supp(s) \subseteq [a,b]$ and s has a continuous distribution function, then for any $\delta \in (0,1]$,*

$$\Pr\left(C_\alpha[s] \leq \widehat{C}_\alpha[s] + (b-a)\sqrt{\frac{5\ln(3/\delta)}{\alpha n}}\right) \geq 1 - \delta,$$

$$\Pr\left(C_\alpha[s] \geq \widehat{C}_\alpha[s] - \frac{b-a}{\alpha}\sqrt{\frac{\ln(1/\delta)}{2n}}\right) \geq 1 - \delta.$$

This lemma tells us that bound the deviation of the empirical CVaR from the true CVaR with high probability. Based on this lemma, we can get

$$\Pr\left(\nu(C_\nu[s(\theta)] - \widehat{C}_\nu[s(\theta)]) \leq \nu(b-a)\sqrt{\frac{5\ln(3/\delta)}{\nu n}}\right) \geq 1 - \delta, \tag{A.1}$$

$$\Pr\left(\nu(C_\nu[s(\theta)] - \widehat{C}_\nu[s(\theta)]) \geq -(b-a)\sqrt{\frac{\ln(1/\delta)}{2n}}\right) \geq 1 - \delta, \tag{A.2}$$

$$\Pr\left(-\mu(C_\mu[s(\theta)] - \widehat{C}_\mu[s(\theta)]) \geq -\mu(b-a)\sqrt{\frac{5\ln(3/\delta)}{\mu n}}\right) \geq 1 - \delta, \tag{A.3}$$

$$\Pr\left(-\mu(C_\mu[s(\theta)] - \widehat{C}_\mu[s(\theta)]) \leq (b-a)\sqrt{\frac{\ln(1/\delta)}{2n}}\right) \geq 1 - \delta. \tag{A.4}$$

Combining Eq.(A.1) and Eq.(A.4), we obtain

$$\Pr\left(\mathcal{L}(f,\lambda,\hat{\lambda}) - \widehat{\mathcal{L}}(f,\lambda,\hat{\lambda}) \leq (b-a)\left[\sqrt{\frac{5\nu\ln(3/\delta)}{n}} + \sqrt{\frac{\ln(1/\delta)}{2n}}\right]\right) \geq 1 - \delta. \tag{A.5}$$

Combining Eq.(A.2) and Eq.(A.3), we obtain

$$\Pr\left(\mathcal{L}(f,\lambda,\hat{\lambda}) - \widehat{\mathcal{L}}(f,\lambda,\hat{\lambda}) \geq -(b-a)\left[\sqrt{\frac{5\mu\ln(3/\delta)}{n}} + \sqrt{\frac{\ln(1/\delta)}{2n}}\right]\right) \geq 1-\delta. \qquad (A.6)$$

Substituting $\nu = k/n$ and $\mu = m/n$ into Eq.(A.5) and Eq.(A.6), we get desired results. ∎

A.6 Proof of Proposition 6

Proof We just need to prove that $\max_{y\notin Y} \theta_y^\top x \geq \theta_{[|Y|+1]}^\top x$. If this is not the case, then for any label $y \notin Y$, then its rank in the ranked list is no more than $|Y| + 2$, then the sum of total number of such labels is not larger than $l - (|Y|+2) + 1 = l - |Y| - 1$. And the total number of labels will be $|Y| + |\{y \notin Y\}| \leq l - 1 \neq l$, which is a contradiction. ∎

A.7 Proof of Lemma 7

Proof

$$\begin{aligned}
\sum_{i=m+1}^{n} s_{[i]} &= \sum_{i=1}^{n} s_i - \sum_{i=1}^{m} s_{[i]} \\
&= \sum_{i=1}^{n} s_i - \min_{\lambda}\left\{m\lambda + \sum_{i=1}^{n}[s_i - \lambda]_+\right\} \\
&= -\min_{\lambda}\left\{-\sum_{i=1}^{n}(s_i - \lambda) - (n-m)\lambda + \sum_{i=1}^{n}[s_i - \lambda]_+\right\}. \\
&= -\min_{\lambda}\left\{-(n-m)\lambda + \sum_{i=1}^{n}[\lambda - s_i]_+\right\} \\
&= \max_{\lambda}\left\{(n-m)\lambda - \sum_{i=1}^{n}[\lambda - s_i]_+\right\}
\end{aligned}$$

The second equation holds because of Lemma A.1. The fourth equation holds because the fact of $[a]_+ - a = [-a]_+$. Define $L(\lambda) = (n-m)\lambda - \sum_{i=1}^{n}[\lambda - s_i]_+$. Let $\lambda = \alpha\lambda_1 + (1-\alpha)\lambda_2$, where $1 \geq \alpha \geq 0$, we have

$$\begin{aligned}
&L(\alpha\lambda_1 + (1-\alpha)\lambda_2) \\
&= \alpha(n-m)\lambda_1 + (1-\alpha)(n-m)\lambda_2 - \sum_{i=1}^{n}[\alpha\lambda_1 + (1-\alpha)\lambda_2 - s_i]_+ \\
&= \alpha(n-m)\lambda_1 + (1-\alpha)(n-m)\lambda_2 - \sum_{i=1}^{m}[\alpha(\lambda_1 - s_i) + (1-\alpha)(\lambda_2 - s_i)]_+ \\
&\geq \alpha(n-m)\lambda_1 + (1-\alpha)(n-m)\lambda_2 - \alpha\sum_{i=1}^{n}[\lambda_1 - s_i]_+ - (1-\alpha)\sum_{i=1}^{n}[\lambda_2 - s_i]_+ \\
&= \alpha L(\lambda_1) + (1-\alpha)L(\lambda_2).
\end{aligned}$$

36

Therefore, $\sum_{i=m+1}^{n} s_{[i]}$ is a concave function. ∎

Appendix B. Additional Experimental Details

B.1 Training Settings on Toy Examples for Aggregate Loss

To reproduce the experimental results of `AoRR` on synthetic data, we provide the details about the settings when we are training the model in Table 11. For example, the learning rate, the number of epochs for the outer loop, and the number of epochs for the inner loop.

Data Sets	Outliers	Logistic loss			Hinge loss		
		LR	# OE	# IE	LR	# OE	# IE
	1	0.01	100	1000	0.01	5	1000
	2	0.01	100	1000	0.01	5	1000
	3	0.01	100	1000	0.01	5	1000
Multi-modal data	4	0.01	100	1000	0.01	5	1000
	5	0.01	100	1000	0.01	5	1000
	10	0.01	100	1000	0.01	5	1000
	20	0.01	100	1000	0.01	5	1000
Imbalanced data	1	0.01	100	1000	0.01	5	1000

LR: Learning Rate, OE: Outer Epochs, IE: Inner epochs

Table 11: `AoRR` settings on toy experiments.

B.2 Training Settings on Real Data Sets for Aggregate Loss

We provide a reference for setting parameters to reproduce our `AoRR` experiments on real data sets. Table 12 contains the settings for individual logistic loss. Table 13 is for individual hinge loss. Table 14 is for multi-class learning.

Data Sets	k	m	C	# Outer epochs	# Inner epochs	Learning rate
Monk	70	20	10^4	5	2000	0.01
Australian	80	3	10^4	10	1000	0.01
Phoneme	1400	100	10^4	10	1000	0.01
Titanic	500	10	10^4	10	1000	0.01
Splice	450	50	10^4	10	1000	0.01

Table 12: `AoRR` settings on real data sets for individual logistic loss.

Data Sets	k	m	C	# Outer epochs	# Inner epochs	Learning rate
Monk	70	45	10^4	5	1000	0.01
Australian	80	3	10^4	5	1000	0.01
Phoneme	1400	410	10^4	10	500	0.01
Titanic	500	10	10^4	5	500	0.01
Splice	450	50	10^4	10	1000	0.01

Table 13: `AoRR` settings on real data sets for individual hinge loss.

Data Sets	Learning rate ("warm-up")	Learning rate ("non-warm-up")	# Outer epochs	# Inner epochs
MNIST	0.4	0.5	20	5000

Table 14: `AoRR` settings on MNIST for multi-class learning.

B.3 Training Settings for Multi-label Learning

The settings for `TKML` on three real data sets are shown in Table 15.

Data Sets	C	#Outer epochs	#Inner epochs	Learning rate
Emotions	10^4	20	1000	0.1
Scene	10^4	20	1000	0.1
Yeast	10^4	20	1000	0.1

Table 15: `TKML` settings on each data set.

B.4 Training Settings for Multi-class Learning

Training settings for the MNIST data set in different noise level can be found in Table 16.

Noise level	#Outer epochs	#Inner epochs	Learning rate
0.2	27	2000	0.1
0.3	25	2000	0.1
0.4	21	2000	0.1

Table 16: `TKML` settings on the MNIST data set in different noise levels.

B.5 Training Settings for `TKML-AoRR` Learning

Training settings for the Yeast data set in different noise level can be found in Table 17.

Noise level	#Epochs	Learning rate
0	1000	0.3
0.1	1000	0.3
0.2	1000	0.3
0.3	1000	0.3

Table 17: `TKML-AoRR` settings on the Yeast data set in different noise levels.

References

Jesús Alcalá-Fdez, Alberto Fernández, Julián Luengo, Joaquín Derrac, Salvador García, Luciano Sánchez, and Francisco Herrera. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, 17, 2011.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Dimitri P Bertsekas. *Control of uncertain systems with a set-membership description of the uncertainty.* PhD thesis, Massachusetts Institute of Technology, 1971.

Sanjay P Bhat and LA Prashanth. Concentration of risk measures: A wasserstein distance approach. In *Advances in Neural Information Processing Systems*, pages 11739–11748, 2019.

Zalán Borsos, Mojmir Mutny, and Andreas Krause. Coresets via bilevel optimization for continual learning and streaming. *Advances in Neural Information Processing Systems*, 33, 2020.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008.

David B Brown. Large deviations bounds for estimating conditional value-at-risk. *Operations Research Letters*, 35(6):722–730, 2007.

Andreas Buja, Werner Stuetzle, and Yi Shen. Loss functions for binary class probability estimation and classification: Structure and applications. *Working draft, November*, 3, 2005.

Xiaojun Chang, Yao-Liang Yu, and Yi Yang. Robust top-k multiclass svm for visual category recognition. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 75–83, 2017.

Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 47–60, 2017.

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of machine learning research*, 2(Dec):265–292, 2001.

Koby Crammer and Yoram Singer. A family of additive online algorithms for category ranking. *Journal of Machine Learning Research*, 3(Feb):1025–1058, 2003.

John M Danskin. *The theory of max-min and its application to weapons allocation problems*, volume 5. Springer Science & Business Media, 2012.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Yanbo Fan, Siwei Lyu, Yiming Ying, and Baogang Hu. Learning with average top-k loss. In *Advances in neural information processing systems*, pages 497–505, 2017.

Yanbo Fan, Baoyuan Wu, Ran He, Bao-Gang Hu, Yong Zhang, and Siwei Lyu. Groupwise ranking loss for multi-label learning. *IEEE Access*, 8:21717–21727, 2020a.

Yanbo Fan, Baoyuan Wu, Ran He, Baogang Hu, Yong Zhang, and Siwei Lyu. Groupwise ranking loss for multi-label learning. *IEEE Access*, to appear, 2020b.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

Shuhei Fujiwara, Akiko Takeda, and Takafumi Kanamori. Dc algorithm for extended robust support vector machine. *Neural computation*, 29(5):1406–1438, 2017.

Spyros Gidaris and Nikos Komodakis. Object detection via a multi-region and semantic segmentation-aware cnn model. In *Proceedings of the IEEE international conference on computer vision*, pages 1134–1142, 2015.

Ran He, Wei-Shi Zheng, and Bao-Gang Hu. Maximum correntropy criterion for robust face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8): 1561–1576, 2010.

Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. In *Advances in neural information processing systems*, pages 10456–10465, 2018.

Shu Hu, Yiming Ying, Xin Wang, and Siwei Lyu. Learning by minimizing the sum of ranked range. *Advances in Neural Information Processing Systems*, 33, 2020.

Shu Hu, Lipeng Ke, Xin Wang, and Siwei Lyu. Tkml-ap: Adversarial attacks to top-k multi-label learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7649–7657, 2021.

Simon Jenni and Paolo Favaro. Deep bilevel learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 618–633, 2018.

Takafumi Kanamori, Shuhei Fujiwara, and Akiko Takeda. Robustness of learning algorithms using hinge loss with outlier indicators. *Neural Networks*, 94:173–191, 2017.

Kenji Kawaguchi and Haihao Lu. Ordered sgd: A new stochastic optimization framework for empirical risk minimization. In *International Conference on Artificial Intelligence and Statistics*, pages 669–679, 2020.

Maksim Lapin, Matthias Hein, and Bernt Schiele. Top-k multiclass svm. In *Advances in Neural Information Processing Systems*, pages 325–333, 2015.

Maksim Lapin, Matthias Hein, and Bernt Schiele. Loss functions for top-k error: Analysis and insights. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1468–1477, 2016.

Maksim Lapin, Matthias Hein, and Bernt Schiele. Analysis and optimization of loss functions for multiclass, top-k, and multilabel classification. *IEEE transactions on pattern analysis and machine intelligence*, 40(7):1533–1554, 2017.

Hoai An Le Thi and Tao Pham Dinh. Dc programming and dca: thirty years of developments. *Mathematical Programming*, 169(1):5–68, 2018.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020a.

Junnan Li, Caiming Xiong, Richard Socher, and Steven Hoi. Towards noise-resistant object detection with noisy annotations. *arXiv preprint arXiv:2003.01285*, 2020b.

Yuncheng Li, Yale Song, and Jiebo Luo. Improving pairwise ranking for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3617–3625, 2017a.

Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1910–1918, 2017b.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

Yi Lin. A note on margin-based loss functions in classification. *Statistics & probability letters*, 68(1):73–82, 2004.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

Weifeng Liu, Puskal P Pokharel, and Jose C Principe. Correntropy: Properties and applications in non-gaussian signal processing. *IEEE Transactions on signal processing*, 55(11): 5286–5298, 2007.

Siwei Lyu, Yanbo Fan, Yiming Ying, and Bao-Gang Hu. Average top-k aggregate loss for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Yifei Ma, Li Li, Xiaolin Huang, and Shuning Wang. Robust support vector machine using least median loss penalty. *IFAC Proceedings Volumes*, 44(1):11208–11213, 2011.

Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Sašo Džeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern recognition*, 45(9): 3084–3104, 2012.

Hamed Masnadi-Shirazi and Nuno Vasconcelos. On the design of loss functions for classification: theory, robustness to outliers, and savageboost. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, pages 1049–1056, 2008.

Feiping Nie, Xiaoqian Wang, and Heng Huang. Multiclass capped lp-norm svm for robust classifications. In *Thirty-First AAAI Conference on Artificial Intelligence (AAAI 2017)*, 2017.

Wlodzimierz Ogryczak and Arie Tamir. Minimizing the sum of the k largest functions in linear time. *Information Processing Letters*, 85(3):117–122, 2003.

Alessandro Ortis, Giovanni Maria Farinella, and Sebastiano Battiato. Predicting social image popularity dynamics at time zero. *IEEE Access*, 7:171691–171706, 2019.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952, 2017.

Bilal Piot, Matthieu Geist, and Olivier Pietquin. Difference of convex functions programming applied to control with expert data. *arXiv preprint arXiv:1606.01128*, 2016.

LA Prashanth, Krishna Jagannathan, and Ravi Kolla. Concentration bounds for cvar estimation: The cases of light-tailed and heavy-tailed distributions. In *International Conference on Machine Learning*, pages 5577–5586. PMLR, 2020.

Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. *arXiv preprint arXiv:1109.5647*, 2011.

Ankit Singh Rawat, Aditya Krishna Menon, Andreas Veit, Felix Yu, Sashank J Reddi, and Sanjiv Kumar. Doubly-stochastic mining for heterogeneous retrieval. *arXiv preprint arXiv:2004.10915*, 2020.

Mark D Reid and Robert C Williamson. Composite binary losses. *Journal of Machine Learning Research*, 11(Sep):2387–2422, 2010.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*, 2018.

Cynthia Rudin. The p-norm push: A simple convex ranking algorithm that concentrates at the top of the list. *Journal of Machine Learning Research*, 10(78):2233–2271, 2009.

Leonard J Savage. Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association*, 66(336):783–801, 1971.

Vatsal Shah, Xiaoxia Wu, and Sujay Sanghavi. Choosing the sample with lowest loss makes sgd robust. In *International Conference on Artificial Intelligence and Statistics*, pages 2120–2130. PMLR, 2020.

Shai Shalev-Shwartz and Yonatan Wexler. Minimizing the maximal loss: How and why. In *ICML*, pages 793–801, 2016.

Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on stochastic programming: modeling and theory*. SIAM, 2014.

Yanyao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In *International Conference on Machine Learning*, pages 5739–5748. PMLR, 2019.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 761–769, 2016.

Emir H Shuford, Arthur Albert, and H Edward Massengill. Admissible probability measurement procedures. *Psychometrika*, 31(2):125–145, 1966.

Nathan Srebro and Ambuj Tewari. Stochastic optimization for machine learning. *ICML Tutorial*, 2010.

Pham Dinh Tao and Le Thi Hoai An. Convex analysis approach to dc programming: theory, algorithms and applications. *Acta mathematica vietnamica*, 22(1):289–355, 1997.

Pham Dinh Tao et al. Algorithms for solving a class of nonconvex optimization problems. methods of subgradients. In *North-Holland Mathematics Studies*, volume 129, pages 249–271. Elsevier, 1986.

Hoai An Le Thi, Hoai Minh Le, Duy Nhat Phan, and Bach Tran. Stochastic dca for the large-sum of non-convex functions problem and its application to group variable selection in classification. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3394–3403. JMLR. org, 2017.

Hoai An Le Thi, Hoai Minh Le, Duy Nhat Phan, and Bach Tran. Stochastic dca for minimizing a large sum of dc functions with application to multi-class logistic regression. *arXiv preprint arXiv:1911.03992*, 2019.

Philip Thomas and Erik Learned-Miller. Concentration inequalities for conditional value at risk. In *International Conference on Machine Learning*, pages 6225–6233. PMLR, 2019.

Nicolas Usunier, David Buffoni, and Patrick Gallinari. Ranking with ordered weighted pairwise classification. In *Proceedings of the 26th annual international conference on machine learning*, pages 1057–1064, 2009.

Vladimir Vapnik. Principles of risk minimization for learning theory. In *Advances in neural information processing systems*, pages 831–838, 1992.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 839–847, 2017.

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 322–330, 2019.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, pages 1954–1963. PMLR, 2015.

Jason Weston, Samy Bengio, and Nicolas Usunier. Wsabie: scaling up to large vocabulary image annotation. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 2764–2770, 2011.

Yichao Wu and Yufeng Liu. Robust truncated hinge loss support vector machines. *Journal of the American Statistical Association*, 102(479):974–983, 2007.

Forest Yang and Sanmi Koyejo. On the consistency of top-k surrogate losses. In *International Conference on Machine Learning*, pages 10727–10735. PMLR, 2020.

Yaoliang Yu, Min Yang, Linli Xu, Martha White, and Dale Schuurmans. Relaxed clipping: A global training method for robust regression and classification. In *NIPS*, pages 2532–2540, 2010.

Min-Ling Zhang and Zhi-Hua Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering*, 18(10):1338–1351, 2006.

Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2013.

Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in neural information processing systems*, pages 8778–8788, 2018.