Constrained Optimization to Improve Critical Rare Classes Performance Within the Top-ranking Part

Yuxin Ying[1], Fuzhen Zhuang[2] (✉), Ziyi Liu[1], Dingyuan Zhu[1], Daixin Wang[1], and Xiaobo Qin[1]

[1] Ant Group {yingyuxin.yyx, kaibu.lzy, dingyuan.zdy, daixin.wdx, kyrie.qxb}@antgroup.com

[2] Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing {zfz20081983}@gmail.com

Abstract. The widespread application of deep learning methods has brought to the challenge of enhancing prediction performance within the highest-score segment of model predictions. In critical domains such as insurance fraud detection and bank cash-out detection, the focus is predominantly on the highest predicted scores, which correspond to high-risk users that need to be intercepted. However, most existing work still focuses on optimizing AUC globally, which often means not being the best within the top-ranking part. Besides, these scenarios often face extreme data imbalance, where the positive samples of interest are in the minority. In this paper, we define the top-ranking optimization problem and propose an Augmented Lagrangian Multiplier method (ALM) based approach to solve it. Specifically, we modify the Discounted Cumulative Gain (DCG) metric to serve as the constraint on top-ranking and add it as the regularization terms to the optimization objective. In addition, to ensure the effectiveness of the regularization term and avoid the overfitting problem, we design a dynamically updated cache mechanism to store the hard samples. Our experimental results on three real-world datasets validate the effectiveness of our proposed method, demonstrating its potential to improve top-ranking prediction performance in imbalanced data settings.

Keywords: Insurance Risk Control· Imbalanced Learning· Top-ranking Optimization.

1 Introduction

In recent years, deep neural networks (DNN) have gradually achieved many successful applications in binary classification problems, e.g. the fraud detection and concept classification scenario [1, 10, 32]. Most of these scenarios focus on optimizing the overall binary classification performance like the Mean Average Precision (mAP) metric and Area Under Curve (AUC) [26, 30]. However, in some practical scenarios such as cash-out fraud detection or insurance fraud detection,

they focus more on accurately identifying the high-risk users ranked at the top so that measures such as user banning or account suspension can be taken against them. In this case, the prediction performance for the remaining users is not as critical, and neither is the specific order of the predicted users ranked at the top. Despite its clear importance, the challenge is further amplified by the fact that there is often a significant class imbalance in these scenarios. The positive samples, which represent the high-risk users we are concerned about, are in the minority. Therefore, how to optimize the binary classification performance of top-ranked users in such imbalanced scenarios is an important problem, which we will refer to as the **top-ranking optimization problem**.

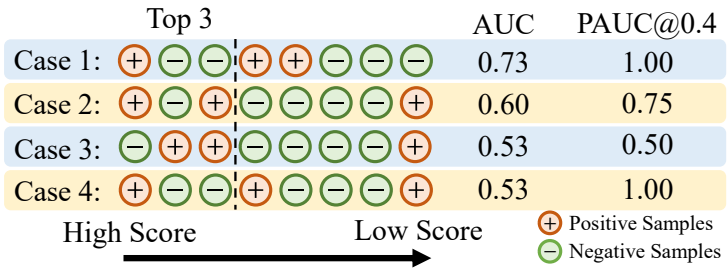

Fig. 1. Four toy examples illustrate the problems with each AUC and PAUC. Case 1 has a higher AUC and PAUC than case 2, while case 2 has a better performance within top 3 part. Cases 3 and 4 have the same AUC, however, the top 3 effect is better in case 3 with a lower PAUC.

To address this problem, some existing works have shed light on different angles. Some methods do not transform the optimization objective and still aim to maximize the global AUC [26, 30]. By using methods such as sampling, weighting, and calibration, they try to enhance the model's focus on the minority classes [9, 23, 19, 17]. However, AUC pays equal attention to samples of all scores and therefore may not be consistent with our goal for top-ranking optimization. For example, consider case 1 and case 2 in Figure 1. When considering the top 3, case 2 is better while having a lower AUC which is opposite to our expectations.

The other work proposes the Partial AUC(PAUC) optimization problem [**?**,12, 29], which refers to targeting the model's top prediction effectiveness by optimizing the AUC score within a certain lower range of the False Positive Rate(FPR). However, PAUC only constrains the FPR, i.e., the top k negative samples, not the top k of the model's predicted scores. This causes it to compare all other samples with the negative sample with the highest predicted score in its computation process, thus over-penalizing some cases with better prediction at the top-ranking part prediction. For example, as case 3 and case 4 in Figure 1. When considering the top 3, case 3 has the better results of having 2 positive samples within. However, the PAUC within $(0, 2/5)$ for both is 0.5 and 1.0 respectively, which is the opposite of what we expected. To sum up, existing methods pay

little attention to targeted top-ranking part optimization. Imbalanced learning and PAUC optimization methods focus on some similar situations but are not identical.

In this paper, we model this class of top-ranking part optimization problems represented by the insurance fraud detection domain. Specifically, we transform the top-ranking part prediction optimization problem into a pairwise optimization problem with constraints using the Augmented Lagrangian Multiplier method. Besides, a modified DCG score of the bipartite ranking task is introduced as a constraint to the regularization term. In addition, to alleviate the problem of positive samples being too sparse, a dynamic cache mechanism is introduced to store the hard positive samples and negative samples. In each round of training, the samples stored in the cache are also added to training to accelerate the convergence of the model. Our proposed method is independent of the specific model structure and can be combined with any binary classification model. We conduct rich experiments on three different types of real-world datasets, and the experimental results show that our method outperforms existing methods for SOTA.

The main contributions of this paper are as follows:

- We propose the **T**op-**R**anking **A**ugmented **L**agrangian method (TRAL) to optimize the top-ranking prediction results of deep learning models in the presence of extreme imbalance of positive and negative samples. The model's focus on positive samples is enhanced by introducing sample difficulty and positive sample ordering.
- We propose a dynamic caching mechanism to store the misclassified positive samples and hard negative samples in each round, thus mitigating the problem of sparse positive samples causing the regularization term to overfit.
- We have conducted sufficient experiments on several real-world datasets, and the extensive experiment results prove the superiority and generalizability of the TRAL proposed in this paper.

2 Preliminaries

2.1 Problem Formulation

In this section, a formal definition of the top-ranking optimization problem in the unbalanced scenario that we try to solve in this paper is given. For an unbalanced binary categorical dataset $D = \{(x_1, y_1), (x_2, y_2), ., (x_n, y_n), y_i \in \{0, 1\}\}$, where x_i is a data sample and y_i is a label. We are concerned with positive samples i.e., y = 1. Then there is a number of negative samples much larger than the number of positive samples. For example, in the insurance fraud detection scenario, normal users account for more than 99% of all users and high-risk fraudulent users account for less than 1%. The prediction result of the model is often used as the blocking threshold in the business. However, the consequences of wrongly intercepting a regular user are much smaller than those of wrongly passing a

high-risk user, so the model's prediction results for positive samples need to be maximized, i.e., the system needs to operate with a high false-negative rate.

Formally, we have $\mathcal{P} = \{x_1^+, \ldots, x_{|\mathcal{P}|}^+\}$ and $\mathcal{N} = \{x_1^-, \ldots, x_{|\mathcal{N}|}^-\}$ representing positive class and negative class respectively. We define the positive class as the critical minority class in our following discussion, i.e., $|\mathcal{P}| << |\mathcal{N}|$. Our goal is to develop a generalized method for inducing a Deep Neural Network(DNN) classifier $f_\theta : \mathcal{R}_d \to \mathcal{R}$ that maps d-dimensional inputs to output scores, thereby boosting the proportion of positive samples within the top k% of the model's predicted scores when sorted in descending order. Let $F(\theta)$ be the total loss function and $f(\theta)$ be the generic loss function for the binary classification problem.

2.2 Augmented Lagrangian Multiplier Method

With $F(\theta)$ representing the total loss function, the optimization problem with constraint $\mathcal{C}(\theta)$ can be formulated as:

$$arg\min_{\theta \in \Theta} F(\theta) \ s.t. \ \mathcal{C}(\theta). \tag{1}$$

Since it is hard to optimize the constrained problem directly, the method of Lagrange Multipliers was introduced to convert the constrained optimization problem to an unconstrained one. The constraint to the objective is added as a normalization part using Lagrange multipliers λ as follows:

$$\mathcal{L}(\theta, \lambda) = F(\theta) + \sum_{i=1}^{m} \lambda_i c_i(\theta). \tag{2}$$

The optimization function obtained after the transformation of the original Lagrange multiplier method [2] is not guaranteed to be smooth and the gradient descent method cannot be used directly because the function is not guaranteed to be strongly convex. By incorporating the quadratic penalty term, the augmented Lagrange method guarantees the smoothness of the optimization process. Thus, the original optimization problem with constraint $\mathcal{C}(\theta)$ can be transformed into the following unconstrained optimization problem, where $\mathcal{C}(\theta)$ remains the constraint of interest:

$$\mathcal{L}_\mu(\theta, \lambda) = F(\theta) + \mu \sum_{i=1}^{m} ||c_i(\theta)||^2 + \sum_{i=1}^{m} \lambda_i c_i(\theta), \tag{3}$$

here μ is a predefined penalty parameter used to control the contribution of the penalty term to the overall loss function.

3 Methods

To solve the top-ranking part optimization problem on imbalanced datasets, we propose a model-independent optimization method with constraints by defining

sample difficulty and designing a dynamic caching mechanism to ensure the validity of the regular terms. First, we discuss how to transform the objective of top-ranking part optimization into a constrained optimization problem and solve it with the Augmented Lagrangian Multiplier method. Second, we use a caching mechanism to temporarily store the positive samples misclassified by the model to alleviate the problem that the regularization term does not work in training due to sparse positive samples.

3.1 Metric for Top-ranking Optimization

For binary classification tasks, several metrics have been proposed for different purposes. A commonly used optimization metric is Area Under Curve(AUC),

$$AUC = \frac{\sum 1(y(x_i^+) > y(x_j^-))}{|\mathcal{N}||\mathcal{P}|} \tag{4}$$

However, since AUC is of equal concern for all score bands, boosting the rankings of both the low-score positive samples and the high-score positive samples will do the same decrease to the loss function. Discounted Cumulative Gain (DCG) is a commonly used ranking metric in top k recommendation tasks [11].

$$DCG@k = \sum_{i=1}^{k} \frac{2^{rel(i)} - 1}{\log_2(i+1)}, \tag{5}$$

where $rel(i)$ denotes the predicted score of the sample in the ith position.

Recall that our goal is to maximize the proportion of positive samples ranked within the top k of predictive scores, so we need to pay attention to the rank order of the positive samples in each prediction. However, the vanilla DCG can be influenced by the ranking within the top-ranking part which is not our primary concern. Therefore, we modified the vanilla DCG metric into the following form:

$$\widehat{DCG} = \sum_{x^+ \in \mathcal{P}} \frac{1}{\log_2(rank(x^+) + 1)}. \tag{6}$$

Since we only have two relationship types for positive and negative samples and do not focus on the rankings within the same class, in the modified $\widehat{DCG}$, we calculate the rank of positive samples by considering neighboring positive samples as tied at the same rank.

3.2 Top-ranking Constrained Optimization

To this end, we define the constrained optimization problem for top optimization as follows:

$$\begin{aligned} &\arg\min F(\theta) \\ &s.t. \sum_{j=1}^{|\mathcal{N}|} \max_{x^+ \in \mathcal{P}} (0, -(f_\theta(x^+) - f_\theta(x_j^-)) + \delta_j) = 0, \end{aligned} \tag{7}$$

where $f_\theta(x)$ represents the predicted score of the DNN model, δ is the classification margin. It is worth noting that δ is not a fixed hyperparameter, but is dynamically adjusted for each negative sample based on its classification difficulty. Satisfying the constraint would be equated to optimizing the modified $\widehat{DCG}$ for binary classification.

We then convert the optimization problem according to the Augmented Lagrangian Method into the following form:

$$\mathcal{L}(\theta, \lambda) = F(\theta) + \frac{\mu \sum_{i=1}^{|\mathcal{P}|} \mathcal{L}_i^2}{2 \cdot |\mathcal{P}| \cdot |\mathcal{N}|} + \frac{\sum_{i=1}^{|\mathcal{P}|} \lambda_i \mathcal{L}_i}{|\mathcal{P}| \cdot |\mathcal{N}|}, \tag{8}$$

where $\mathcal{L}_i = \sum_{j=1}^{|\mathcal{N}|} \max(0, -(f_\theta(x_i^+) - f_\theta(x_j^-)) + \delta_{ij})$. Note that the δ_{ij} is different with each sample pair. Unlike previous work ALM, our δ_{ij} here is not a fixed hyper-parameter set in advance, but a learnable parameter related to the classification difficulty of each negative sample.

This function can't be used as an optimization objective yet, because the $\max(\cdot)$ function in the $\mathcal{L}_i$ calculation is non-continuous. Therefore, we use a widely-used surrogate function for $\max(\cdot)$ to convert it into a continuous convex function, i.e. $max(x_1, \ldots, x_n) \approx \log(\sum_{i=1}^{n} \exp(x_i))$.

$$\mathcal{L}_i = -\log\left(\frac{e^{f(x_i^+)}}{e^{f(x_i^+)} + |\mathcal{N}| \sum_{j=1}^{|\mathcal{N}|} \eta_{ij} e^{f(x_j^-)}}\right), \tag{9}$$

where $\eta_{ij} = \frac{exp(\delta_{ij})}{|\mathcal{N}|}$ is defined as the classification difficulty of negative samples.

It is not difficult to prove that $\mathcal{L}_i$ restricts the upper and lower bounds of the $\widehat{DCG}$, and optimizing $\mathcal{L}_i$ results in a bounded optimization of the $\widehat{DCG}$. For the proof of the upper and lower bounds of the $\widehat{DCG}$ one can refer to the Proof section in the Appendix. After obtaining that $\mathcal{L}_i$ is the equivalent boundary of the $\widehat{DCG}$, the optimization objective is equivalent to optimizing the AUC while satisfying the constraints of the $\widehat{DCG}$ since we use the ALM transformation constraints to add them to the optimization function as regular terms.

It is worth noting that since $\delta \in [0, 1]$, the value of η belongs to $[\frac{1}{|\mathcal{N}|}, \frac{e}{|\mathcal{N}|}]$. And η is inversely proportional to δ. When the classification difficulty of negative samples is lower, the classification margin with positive samples is larger, resulting in a smaller η, i.e., the impact on the optimization term. Therefore, we can extract η as a measure of the classification difficulty of negative samples.

3.3 Dynamic Cache Module

Under the imbalanced scenario, due to the extreme sparsity of positive samples, there may not always be enough positive samples within each mini-batch to participate in training. This leads to the constraint term in equation (8) not working. To solve this problem, we borrow the caching mechanism from the computer hardware field to ensure that the model can see enough positive samples. Specifically, we use an LRU-like cache mechanism to maintain a queue of positive samples of size q and concate them into each mini-batch during training.

First, before the training process begins, we randomly select q positive samples to initialize the cache. Subsequently, after each training step, we input all the samples within this batch and the cache to the model for prediction and pick the ones in which the model predicts incorrectly (i.e., those with prediction scores of 0.5 or less). Then we sort the prediction scores decreasingly and select the lowest q samples to update the cache.

Such a simple method may be effective at the beginning of the training, however, as the number of iterations increases, the frequency of updating the samples in the cache will gradually decrease, leading to a serious over-fitting of those samples. This is supported by some of our preliminary experimental results. Therefore, we also design two mechanisms to alleviate the over-fitting problem. First, we modify the update rule for positive samples. Specifically, if the model correctly predicts a positive sample it is moved out of the cache and marked as the correct answer. Second, we also incorporate the idea of sampling hard negative samples. As the model's effectiveness improves, the number of positive samples incorrectly predicted in each round gradually decreases to below q. This leads to a reduction in the number of samples in the cache. However, it is important to realize that this does not mean the model predicts well enough. It is just that due to the presence of the ALM regularization term and the cache module, we will inevitably cause a rise in the mean value of the model prediction scores. This means that the predictive scores of many negative samples are also rising. Therefore, to suppress the negative sample prediction scores and prevent the cache module from causing over-fitting problems, we performed dynamic sampling based on the classification difficulty of the negative samples. Each negative sample's probability of being sampled is then determined by the difficulty η described above.

$$p_j = \begin{cases} \eta_j = \sum_{i=1}^{|\mathcal{P}|} \frac{exp(\delta_{ij})}{|\mathcal{N}|} & , j \in \mathcal{N}[1,k] \\ 0 & , j \in others \end{cases}, \tag{10}$$

where $\mathcal{N}[1,k]$ represent the top-ranked $k\%$ negative items. It can be inferred that the greater the δ is, the harder the samples will be drawn. The core idea is that the more difficult samples are sampled, the higher the probability that they will enter the cache, which always maintains samples on the model prediction boundaries so that the regularization term always plays a role.

4 Experiments

In this section, we present our experimental evaluation of the top-ranking part optimization task under imbalanced datasets. To verify the generalizability of our approach, we conducted experiments on three datasets with different task scenarios but with similar distributions: the Insurance dataset obtained from a widely used online Health Insurance platform, the public dataset Fraud of credit card fraud detection task, and the Criteo recommendation dataset. On

Table 1. Statistics of three large-scale datasets. The positive ratio indicates the proportion of positive samples.

dataset	pos	neg	positive ratio
Insurance	99,386	1,012,716	0.99%
Fraud	492	284,807	0.17%
Criteo	7,450	745,051	1.00%

the Insurance control scenario and Fraud detection dataset, we choose MLP as the benchmark classification model, and on the recommendation dataset Criteo we choose DeepFM [8] as the benchmark model.

4.1 Datasets

Insurance dataset contains users' features such as basic information and medical records of the insured users, labeled as whether they are insured or not. This task focuses on predicting whether a user is likely to be insured during the policy period based on the user's characteristics, and the high-risk users predicted by the model need to be blocked or otherwise dealt with to reduce the risk to the insurance company. This dataset is collected from the history of the online business within 30 days. The ratio of positive samples (insured users) and negative samples (regular users) is about 1:100. We divide the train, validation, and test set according to the ratio of 8:1:1. The first 26 days of data are selected as training data and the last two days as the test set.

Fraud dataset contains the transaction history of credit card holders in Europe for September 2013. The task focuses on predicting the presence of possible credit card fraud based on a user's transaction history. The dataset contains 284,807 transactions, of which 492 are fraudulent. We divide the training, validation, and testing datasets randomly according to the ratio of 8:1:1.

Criteo dataset contains data on user ad clicks over 7 days from the CriteoLabs website. This task focuses on predicting the click-through rate of users clicking on displayed advertisements based on user information and information about the currently visited page. We randomly sampled the clicks and hit an overall positive-to-negative sample ratio of 1:100. We followed the original train and test split.

4.2 Baselines

We compare the proposed method with the following competitive and mainstream methods which aim to improve the model's performance over the top-ranking part. In addition, we have chosen Partial AUC optimization methods as a comparison of direct AUC optimization methods such as mini-batch AUC(MBAUC) and SPAUCI methods.

- ALM [19] first introduced the constraint optimization problem to enforce maximal AUC through prioritizing FPR reduction at high TPR.
- RankReg [17] add a ranking-based regularization term to improve TPR while reducing FPR.
- MBAUC [7] leverage the direct optimization of AUC for binary classification problems.
- SPAUCI [22] proposed a non-convex strongly concave min-max regularized problem of instance-wise loss functions for PAUC optimization.

Besides, following previous work [19, 17], we consider applying regularizer-based methods with several widely-used cost-sensitive loss functions: Cost-Weighted Binary Cross Entropy loss(WBCE) [30], Symmetric Marginal Loss(S-ML) [15], Symmetric Focal Loss(S-FL) [14], and Label Distribution Perceived Marginal Loss(LDAM) [4].

4.3 Implemetation Details

For the binary imbalanced classification dataset Insurance and Fraud, we adopt MLP as the backbone architecture with shape **[512, 2]**. As for the CTR prediction task of Criteo, we choose DeepFM [8] as the backbone architecture with the embedding dimension of **32**. Throughout all the experiments, we set the batch size to 2048 and used the Adam optimizer.

4.4 Evaluation Metrics

To evaluate the overall classification performance of our proposed method and the baselines described above, we follow the existing works to use the standard metric AUC. Besides, in risk-concern imbalance learning scenarios, the minority class of high-risk positive samples is our primary concern. Therefore, we adopt PAUC@k and Prec@k metrics to evaluate the performance within the top-ranking part. In practice, we focus on the top 3% part, but we also focus on some specific percentage on the top to further investigate the influence of our proposed method, i.e., top 1%, 2%, 3%, 4%, 5%, respectively. For all experiments, we report the results with 95% confidence intervals on the average of 5 runs.

4.5 Main Result

As shown in Table 2, we can observe that the proposed method outperforms the baseline models on different datasets and losses. Grouped by base loss, it can be seen that our proposed TRAL reach the best performance within each group. This shows that applying this model to other loss functions as well as the base model can be improved, reflecting the generalizability of this model.

For the Insurance dataset, it is clear that the proposed TRAL is consistently better on most metrics. Comparing the results within each block, we can see that the method in this paper achieves a significant improvement over both ALM

Table 2. Model comparison on three real-world datasets. We record the mean results over 5 runs. * indicates a significant improvement compared with the best baseline ($p < 0.05$ on paired t-test).

	Insurance			Fraud			Criteo		
Methods	auc	pauc@3%	prec@3%	auc	pauc@3%	prec@3%	auc	pauc@3%	prec@3%
MBAUC	0.5417	0.3215	1.5410	0.8234	0.6215	36.45	0.7255	0.0829	4.512
SPAUCI	0.5701	0.3417	1.5787	0.8248	**0.6473**	37.19	0.7215	**0.0883**	4.676
BCE	0.5516	0.2987	1.5085	0.8257	0.6061	30.80	0.7029	0.0609	3.619
+ALM	0.5488	0.3025	1.5117	0.8226	0.5996	34.54	0.7056	0.0787	4.560
+RankReg	0.5526	0.3091	1.5245	0.8267	0.6117	35.61	0.7080	0.0781	4.509
+TRAL	0.5548	0.3157	1.5469	0.8363	0.6209	37.47	0.7133	0.0812	4.594
WBCE	0.5523	0.2991	1.5236	0.8261	0.6072	31.23	0.7034	0.0678	3.754
+ALM	0.5531	0.3029	1.5145	0.8297	0.6092	34.37	0.7067	0.0814	4.494
+RankReg	0.5512	0.3061	1.5164	0.8289	0.6107	34.79	0.7045	0.0817	4.531
+TRAL	0.5587	0.3154	1.5457	0.8324	0.6217	38.32	0.7157	0.0823	4.572
S-ML	0.5547	0.3107	1.5577	0.8334	0.6125	35.61	0.7131	0.0726	3.975
+ALM	0.5619	0.3201	1.5684	0.8418	0.6217	36.24	0.7191	0.0826	4.561
+RankReg	0.5642	0.3217	1.5687	0.8416	0.6231	36.17	0.7201	0.0831	4.562
+TRAL	0.5701	0.3301	1.5774	0.8501	0.6314	37.24	0.7295	0.0873	4.662
S-FL	0.5551	0.3312	1.5578	0.8350	0.6157	35.17	0.7143	0.0721	4.013
+ALM	0.5621	0.3314	1.5664	0.8421	0.6234	36.41	0.7221	0.0832	4.570
+RankReg	0.5627	0.3320	1.5658	0.8427	0.6238	36.50	0.7225	0.0831	4.579
+TRAL	0.5710	0.3397	1.5780	0.8523	0.6327	37.36	0.7237	0.0845	**4.842***
LDAM	0.5567	0.3340	1.5601	0.8352	0.6206	35.24	0.7165	0.0743	4.102
+ALM	0.5634	0.3407	1.5721	0.8435	0.6301	36.67	0.7251	0.0841	4.617
+RankReg	0.5629	0.3398	1.5717	0.8437	0.6311	36.71	0.7246	0.0847	4.621
+TRAL	**0.5714***	**0.3421***	**1.5801**	**0.8543***	0.6424	**38.47***	**0.7307***	0.0871	4.836

and RankReg. Comparing the different loss functions vertically, we can see that our method achieves relatively best results in each metric when combined with LDAM.

Similar results can be observed on the Fraud and Criteo dataset except for pauc@3%, where SPAUCI has the best score. This is because SPAUCI is specifically optimized for Partial-AUC. However, as mentioned earlier, pauc@3% can only reflect the model's ability to categorize and identify positive samples within the top-ranking part to a certain extent. In addition, the actual training speed and convergence speed of SPAUCI method is prolonged, which is not practical for the actual data scale faced by the industry.

Note that our optimization goal in this task is the performance of the model in the part with the highest prediction scores, so the global AUC metric is a reference metric for us rather than an optimization focus of interest. However, we still achieve high AUC scores in many experiments, which shows that our approach not only improves the model's ability to recognize top-ranking samples but also enhances the model's global classification ability at the same time.

In addition, unlike the RankReg method that introduces the rank index of positive samples as a regularization term, our proposed TRAL is mainly based on the ALM method and introduces the definition of negative sample difficulty to measure the effect of the sample on the regularization term. Therefore, it can be seen that the enhancement of the proposed method in this paper is the most significant on the Fraud dataset. This is because the Fraud dataset has relatively the least difficult data and the largest number of simple negative samples among the three datasets. Vanilla classification methods can quickly obtain a high AUC

Table 3. Ablation Study of different update strategies for cache module on Insurance datasets.

	Insurance		
Methods	AUC	PAUC@3%	Prec@3%
TRAL	**0.5714**	**0.3421**	**1.5801**
w/o cache	0.5571	0.3341	1.5617
w/o hns	0.5412	0.3217	1.5210
threshold	0.5532	0.3331	1.5512
percent	0.5545	0.3315	1.5524
baseline	0.5567	0.3340	1.5601

score on this dataset, however, to further improve the model's prediction ability for the top-ranking samples will soon face a serious overfitting problem. The proposed TRAL, however, combines the difficulty of negative samples with the ALM method to make the regularization term work consistently, which further improves the model's performance.

4.6 Ablation Study

To visually verify the usefulness of the various components of the proposed TRAL in this paper, we performed ablation experiments on the Insurance dataset. We chose to use the TRAL combined with WBCE loss as the baseline model. Based on this, we do the following for the cache module and the computation of the regularization term, respectively: (1)w/o cache: remove the cache module, (2)w/o hns: remove the hard negative sample strategy and store the cache model for positive samples only, (3)threshold: update the cache with the k positive samples with the lowest scores within each batch, (4)percent: update the cache with the k% positive samples with the lowest scores within each batch, (5)baseline: the baseline MLP model with LDAM loss.

As can be seen from Table 3, the decrease in the effectiveness of the models with the corresponding modules removed is very significant. The removal of the cache module directly decreases the effectiveness of the model by **2.5% points**. It is worth noting that, the method of removing the dynamic negative sampling mechanism is less effective than simply removing the entire cache module. This implies that if a cache module is added directly, the effect on the model prediction results is likely to be negative. In our experiments, we found that after a certain number of iterations, the number of incorrectly predicted positive samples in each round of training is less than the pre-set cache size, which leads to a portion of the samples in the cache not being updated. These "stubborn" positive samples cause the model to overfit, and thus worsen the model's prediction effect.

4.7 Parameter Analysis

In the main experiment, we chose the top 3% as the main criterion to evaluate the model effect, which is based on the actual scenario needs of our online business

Table 4. Illustration of the performance of the top-ranking part over different ratio. * indicates a significant improvement compared with the best baseline ($p < 0.05$ on paired t-test).

	Prec				
Methods	@1%	@2%	@3%	@4%	@5%
BCE	4.1150	3.8671	3.6192	3.4205	3.0239
ALM	6.3956	5.2553	4.5603	4.1517	3.9956
RankReg	5.5431	4.7230	4.5094	4.3210	4.0479
TRAL	**7.0897***	**5.4536**	**4.5945***	4.2880	4.0452
MBAUC	5.5912	5.1250	4.5120	4.1457	4.0537
SPAUCI	6.6931	5.3793	4.5760	**4.3500**	**4.0650**

and the empirical scores from the past analysis. In order to further explore the impact of the proposed methodology on the model prediction scores in different degrees, we explored the variation of the prediction effects in different scales from top 1% to 5%. In Table 4, we list the corresponding experimental results with different percentages. Among them, for the PAUC optimization methods, we set their FPR objectives upper bounds to the corresponding percentages as well to obtain an approximate top-ranking optimization objective.

Compared to the PAUC class method, the improvement of TRAL is more significant in scenarios with smaller top ratios. This result is in line with our expectations. As mentioned earlier, the PAUC class of optimization methods may penalize some results that are predicted accurately in the top-ranking part because they focus on the model's ability to classify the positive samples with the highest scores rather than the positive samples within a certain portion of the highest predicted scores.

5 Related Works

5.1 Imbalance Learning.

Existing imbalance learning methods can be mainly categorized into three main types: pre-processing, mid-processing, and post-processing according to their action stages. Pre-processing methods mainly act in the data processing stage, changing the data distribution through resampling or data augmentation to increase the number of minority class samples and mitigate the imbalance problem [27, 5, 31]. Post-processing methods, on the other hand, aim to correct the model's inherent bias towards the majority class by utilizing techniques such as calibrating the model's predictive distribution based on the data distribution. Mid-processing methods mainly rely on designing category-sensitive loss functions to modify the optimization objective during the training process [30, 23, 9, 24, 13]. For example, Weighted Binary Cross-Entropy(W-BCE) increases the impact of minority category samples on the loss function by multiplying their contribution by a scaling factor [30]. Other methods, such as Symmetric

Marginal Loss [15] and symmetric Focal Loss [14], introduce margin-based penalties to enhance the separation of class decision boundaries. In addition, Class Balanced BCE [6]and Label Distribution-Aware Margins [4] address the imbalance problem by inversely weighting the loss according to the class frequency or minimizing the margin-based generalization boundaries.

Additionally, some studies focus on extreme multi-label classification (XMLC), aiming to address the optimization issues of long-tail classes when multi-class models are dominated by mainstream samples. Since Bhatia [3] proposed the XC benchmark dataset in 2016, researchers have made notable progress. Schultheis [21] introduced an expected test utility (ETU) framework to optimize generalized at-k metrics, tackling long-tail label challenges by deriving optimal prediction rules and efficient approximations with regret guarantees. Later, they advances the algorithm by developing a consistent Frank-Wolfe algorithm for complex macro-at-k metrics within the population utility framework and transforming classifier optimization into confusion matrix optimization to address budgeted predictions at k in multi-label classification. [20] .

The method proposed in this paper mainly belongs to the category-sensitive loss in the mid-processing approach and is combined with the model-based approach. We optimize the regular term based mainly on the work of ALM [19], so that the loss function can focus more on the prediction effect of the model on the top-ranking samples, and ensure the concentration of the positive samples through the dynamic caching mechanism introduced in this paper so that the regular term can work continuously.

5.2 Partial-AUC Optimization.

The concept of partial-AUC was initially introduced by [16], with initial research efforts primarily concentrated on its application to straightforward linear models. A distribution-free, rank-based method was employed for the first time to optimize PAUC in a seminal work [16]. Another study [25] focused on the non-parametric estimation of PAUC and incorporated feature selection iteratively to construct the ultimate classifier model. A more sophisticated approach [18] was later introduced, where a cutting-plane algorithm was developed to identify the instances that most significantly violated the constraints. This method broke down the PAUC optimization into several smaller problems, which were then addressed through a structured SVM-based technique. Despite these advancements, many of the existing methods faced challenges such as a lack of differentiation properties or intractable optimization issues. To address this, an implicit function theorem was introduced to formulate a rate-constrained optimization problem that treated the quantization threshold as a function of the model's parameters [12]. A recent study by [28] made significant strides in enabling end-to-end optimization of PAUC in deep learning models. This was achieved by streamlining the complex sample selection process inherent in PAUC optimization into a two-tiered optimization strategy. The inner layer was dedicated to selecting instances, while the outer layer focused on minimizing the loss function. Despite this progress, these methods were subject to approximation errors when

estimating the actual PAUC. Subsequent research in [29, 33] introduced a PAUC estimator with smooth properties, offering theoretical assurances of convergence for their algorithm. Nevertheless, the utility of this algorithm was constrained by its slow rate of convergence, particularly when applied to the Two-way PAUC.

6 Conclusion

The prevalent utilization of deep learning in data mining has limitations on imbalanced datasets, where the minority class, often of primary interest, is underrepresented and less effectively modeled compared to the majority class. This paper introduces an Augmented Lagrangian method base optimization transformation that prioritizes the model's prediction within the top-ranking part. The proposed TRAL method leverages the modified DCG metric's ranking properties to enhance top prediction effectiveness, incorporating these as regularization terms within the optimization function. Furthermore, to stabilize regularization against the challenge of scarce positive samples, we have defined the concept of sample difficulty and developed a Dynamic Cache mechanism, thereby improving the model's accuracy in top-ranking predictions, which is particularly relevant in risk-aware domains such as insurance and fraud detection.

Acknowledgments. This research work is supported by the National Key Research and Development Program of China under Grant NO. 2024YFF0729003, the National Natural Science Foundation of China under Grant NOs. 62176014, the Fundamental Research Funds for the Central Universities, Ant Group Research Fund.

References

1. Almarshad, F.A., Gashgari, G.A., Alzahrani, A.I.A.: Generative adversarial networks-based novel approach for fraud detection for the european cardholders 2013 dataset. IEEE Access **11**, 107348–107368 (2023)
2. Bertsekas, D.P.: Multiplier methods: A survey. Autom. **12**(2), 133–145 (1976)
3. Bhatia, K., Dahiya, K., Jain, H., Kar, P., Mittal, A., Prabhu, Y., Varma, M.: The extreme classification repository: Multi-label datasets and code (2016), http://manikvarma.org/downloads/XC/XMLRepository.html
4. Cao, K., Wei, C., Gaidon, A., Aréchiga, N., Ma, T.: Learning imbalanced datasets with label-distribution-aware margin loss. In: NeurIPS. pp. 1565–1576 (2019)
5. Chou, H., Chang, S., Pan, J., Wei, W., Juan, D.: Remix: Rebalanced mixup. In: ECCV Workshops (6). Lecture Notes in Computer Science, vol. 12540, pp. 95–110. Springer (2020)
6. Cui, Y., Jia, M., Lin, T., Song, Y., Belongie, S.J.: Class-balanced loss based on effective number of samples. In: CVPR. pp. 9268–9277. Computer Vision Foundation / IEEE (2019)
7. Gultekin, S., Saha, A., Ratnaparkhi, A., Paisley, J.W.: MBA: mini-batch AUC optimization. IEEE Trans. Neural Networks Learn. Syst. **31**(12), 5561–5574 (2020)
8. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: Deepfm: A factorization-machine based neural network for CTR prediction pp. 1725–1731 (2017)

9. Huang, C., Li, Y., Loy, C.C., Tang, X.: Learning deep representation for imbalanced classification. In: CVPR. pp. 5375–5384. IEEE Computer Society (2016)
10. Ibomoiye, D.M., Sun, Y.: A deep learning ensemble with data resampling for credit card fraud detection. IEEE Access **11**, 30628–30638 (2023)
11. Järvelin, K., Kekäläinen, J.: IR evaluation methods for retrieving highly relevant documents. In: Yannakoudakis, E.J., Belkin, N.J., Ingwersen, P., Leong, M. (eds.) SIGIR 2000: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 24-28, 2000, Athens, Greece. pp. 41–48. ACM (2000). https://doi.org/10.1145/345508.345545, https://doi.org/10.1145/345508.345545
12. Kumar, A., Narasimhan, H., Cotter, A.: Implicit rate-constrained optimization of non-decomposable objectives. In: ICML. Proceedings of Machine Learning Research, vol. 139, pp. 5861–5871. PMLR (2021)
13. Li, Z., Kamnitsas, K., Glocker, B.: Overfitting of neural nets under class imbalance: Analysis and improvements for segmentation. In: MICCAI (3). Lecture Notes in Computer Science, vol. 11766, pp. 402–410. Springer (2019)
14. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV. pp. 2999–3007. IEEE Computer Society (2017)
15. Liu, W., Wen, Y., Yu, Z., Yang, M.: Large-margin softmax loss for convolutional neural networks. In: ICML. JMLR Workshop and Conference Proceedings, vol. 48, pp. 507–516. JMLR.org (2016)
16. McClish, D.K.: Analyzing a portion of the roc curve. Medical decision making **9**(3), 190–195 (1989)
17. Mohammadi, K., Zhao, H., Zhai, M., Tung, F.: Ranking regularization for critical rare classes: Minimizing false positives at a high true positive rate. In: CVPR. pp. 15783–15792. IEEE (2023)
18. Narasimhan, H., Agarwal, S.: A structural SVM based approach for optimizing partial AUC. In: ICML (1). JMLR Workshop and Conference Proceedings, vol. 28, pp. 516–524. JMLR.org (2013)
19. Sangalli, S., Erdil, E., Hötker, A.M., Donati, O., Konukoglu, E.: Constrained optimization to train neural networks on critical and under-represented classes. In: NeurIPS. pp. 25400–25411 (2021)
20. Schultheis, E., Kotłowski, W., Wydmuch, M., Babbar, R., Borman, S., Dembczyński, K.: Consistent algorithms for multi-label classification with macro-at-k metrics. In: 12th International Conference on Learning Representations (ICLR 2024). Curran Associates Inc., United States (2024)
21. Schultheis, E., Wydmuch, M., Kotlowski, W., Babbar, R., Dembczynski, K.: Generalized test utilities for long-tail performance in extreme multi-label classification. In: Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (eds.) Advances in Neural Information Processing Systems. vol. 36, pp. 22269–22303. Curran Associates, Inc. (2023)
22. Shao, H., Xu, Q., Yang, Z., Bao, S., Huang, Q.: Asymptotically unbiased instance-wise regularized partial AUC optimization: Theory and algorithm. In: NeurIPS (2022)
23. Thai-Nghe, N., Gantner, Z., Schmidt-Thieme, L.: Cost-sensitive learning methods for imbalanced data. In: IJCNN. pp. 1–8. IEEE (2010)
24. Wang, S., Liu, W., Wu, J., Cao, L., Meng, Q., Kennedy, P.J.: Training deep neural networks on imbalanced data sets. In: IJCNN. pp. 4368–4374. IEEE (2016)
25. Wang, Z., Chang, Y.I.: Marker selection via maximizing the partial area under the roc curve of linear risk scores. Biostatistics **12 2**, 369–85 (2011)

26. Wei, J., Wang, S., Huang, Q.: F^3net: Fusion, feedback and focus for salient object detection. In: AAAI. pp. 12321–12328. AAAI Press (2020)
27. Xu, Z., Chai, Z., Yuan, C.: Towards calibrated model for long-tailed visual recognition from prior perspective. In: NeurIPS. pp. 7139–7152 (2021)
28. Yang, Z., Xu, Q., Bao, S., He, Y., Cao, X., Huang, Q.: When all we need is a piece of the pie: A generic framework for optimizing two-way partial AUC. In: ICML. Proceedings of Machine Learning Research, vol. 139, pp. 11820–11829. PMLR (2021)
29. Yao, Y., Lin, Q., Yang, T.: Large-scale optimization of partial AUC in a range of false positive rates. In: NeurIPS (2022)
30. Zadrozny, B., Langford, J., Abe, N.: Cost-sensitive learning by cost-proportionate example weighting. In: ICDM. p. 435. IEEE Computer Society (2003)
31. Zhang, H., Cissé, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: ICLR (Poster). OpenReview.net (2018)
32. Zhang, R., Cheng, D., Yang, J., Ouyang, Y., Wu, X., Zheng, Y., Jiang, C.: Pre-trained online contrastive learning for insurance fraud detection. In: AAAI. pp. 22511–22519. AAAI Press (2024)
33. Zhu, D., Li, G., Wang, B., Wu, X., Yang, T.: When AUC meets DRO: optimizing partial AUC for deep learning with non-convex convergence guarantee. In: ICML. Proceedings of Machine Learning Research, vol. 162, pp. 27548–27573. PMLR (2022)

A Appendix

A.1 Proof of the equivalence of $\mathcal{L}_i$ and $\widehat{DCG}$

In this part, we focus on proving that the optimization target above equals optimizing the top-ranking part's performance.

Lemma 1. *Optimizing Equation9 is equivalent to optimizing the upper bound of* $\widehat{DCG}$.

Proof. Consider that for a positive sample x_i^+, its rank order $rank(x_i^+)$ represents that $rank(x_i^+)$ negative samples can have a higher score than x_i^+. Therefore,

$$rank(x_i^+) = 1 + \sum_{j \in \mathcal{N}} sign(f(x_j^-) - f(x_i^+) + \delta_j > 0), \tag{11}$$

and by $sign(x > 0) \leq e^x$, we have

$$rank(x_i^+) \leq 1 + \sum_{j \in \mathcal{N}} e^{f(x_j^-) - f(x_i^+) + \delta_j}. \tag{12}$$

The upper bound of $\widehat{DCG}$ can be achieved as follows.

$$\begin{aligned} \widehat{DCG} &= \sum_{i \in \mathcal{P}} \frac{1}{\log_2 (rank(i) + 1)} \leq \sum_{i \in \mathcal{P}} \log(|\mathcal{N}| e^{-1} + 1) \\ &\leq \sum_{i \in \mathcal{P}} \log(\sum_{j \in \mathcal{N}} e^{f(x_j^-)} - e^{f(x_i^+) + \delta_j} + 1) \\ &= \sum_{i \in \mathcal{P}} -\log \frac{e^{f(x_i^+)}}{e^{f(x_i^+)} + \sum_{j \in \mathcal{N}} e^{f(x_j^-)} + \delta_j} = \mathcal{L}_i. \end{aligned} \tag{13}$$

Lemma 2. *Optimizing Equation 9 is equivalent to optimizing the lower bound of* $\widehat{DCG}$

Proof. The lower bound of $\widehat{DCG}$ can be achieved by the following process.

$$\begin{aligned} \widehat{DCG} &= \exp(\log(\sum_{i \in \mathcal{P}} \frac{1}{\log_2(rank(i) + 1)})) \\ &\geq \exp(\sum_{i \in \mathcal{P}} \frac{1}{rank(i)}) \\ &\geq \exp(\sum_{i \in \mathcal{P}} \frac{1}{\sum_{k \in \mathcal{N}} e^{f(x_k) - f(x_i^+) + \delta_k} + 1}) \\ &= e^{-\mathcal{L}_i}. \end{aligned} \tag{14}$$

By Lemma 1 and Lemma 2, we have

$$e^{-\mathcal{L}_i} \leq \widehat{DCG} \leq \mathcal{L}_i. \tag{15}$$

Consequently, minimizing q is equivalent to minimizing $\widehat{DCG}$, and further, by constraining the number of top k samples for computing the $\widehat{DCG}$, we can explicitly optimize the top-ranking prediction performance.