Minority Disk Failure Prediction Based on Transfer Learning in Large Data Centers of Heterogeneous Disk Systems

Ji Zhang, Ke Zhou, *Member, IEEE*, Ping Huang, Xubin He, *Senior Member, IEEE*, Ming Xie, Bin Cheng, Yongguang Ji, and Yinhu Wang

Abstract—The storage system in large scale data centers is typically built upon thousands or even millions of disks, where disk failures constantly happen. A disk failure could lead to serious data loss and thus system unavailability or even catastrophic consequences if the lost data cannot be recovered. While replication and erasure coding techniques have been widely deployed to guarantee storage availability and reliability, disk failure prediction is gaining popularity as it has the potential to prevent disk failures from occurring in the first place. Recent trends have turned toward applying machine learning approaches based on disk SMART attributes for disk failure predictions. However, traditional machine learning (ML) approaches require a large set of training data in order to deliver good predictive performance. In large-scale storage systems, new disks enter gradually to augment the storage capacity or to replace failed disks, leading storage systems to consist of small amounts of new disks from different vendors and/or different models from the same vendor as time goes on. We refer to this relatively small amount of disks as minority disks. Due to the lack of sufficient training data, traditional ML approaches fail to deliver satisfactory predictive performance in evolving storage systems which consist of heterogeneous minority disks. To address this challenge and improve the predictive performance for minority disks in large data centers, we propose a minority disk failure prediction model named *TLDFP* based on a transfer learning approach. Our evaluation results in two realistic datasets have demonstrated that *TLDFP* can deliver much more precise results and lower additional maintenance cost, compared to four popular prediction models based on traditional ML algorithms and two state-of-the-art transfer learning methods.

Index Terms—Disk failure, machine learning, transfer learning, cloud computing, data center

1 Introduction

Hard disks are widely used as the common and primary storage devices for large-scale storage systems in modern data centers. In such data centers, it has been an extremely challenging undertake to ensure high availability and reliability for IT management, as various disk failures constantly occur in the field, whether being hard disks [2], [3], [4], flash-based SSDs [5], [6] or NVMes. Disk failures can lead to temporary data loss and thus system unavailability or even permanent data loss if the lost data cannot be recovered by existing data protection schemes, e.g., replication and erasure codes [7], [8] due to disk failures exceeding the designed correction capability. A disk is a rather complex system consisting of a variety of magnetic, mechanical, and electronic components, each of which could fail. As a result, disk failures show different manifestations and extents of severeness [9] for numerous reasons, which has been observed in data centers from major IT companies [10], [11]. Compared with the traditional passive fault tolerance techniques like Erasure Code (EC) and Redundant Arrays of Independent Disks (RAID) [12], proactive disk failure prediction tends to ensure the reliability and availability of large-scale storage systems in advance. Therefore, successful disk failure prediction not only reduces the risk of losing data but also reduces the data recovery cost (i.e., network bandwidth) associated with recovering the data residing on failed disks.

Disk manufacturers implement the self-monitoring, analysis and reporting technology (SMART) technology [13] in the disk firmware. Most of the SMART attributes contain information about gradual degradations and possible defects of disks. Internally, a disk uses the so-called *"threshold method"* [14] based on SMART values to claim its failure status, which means the hard disk would raise an alarm if the value of an SMART attribute crosses the corresponding predefined threshold. However, this *"threshold method"* only achieves a failure detection rate (FDR) of 3-10 percent with 0.1 percent false alarm rate (FAR) [14] (FDR and FAR see Section 5.1.2 in detail). In other words, these numbers highlight the

- *J. Zhang and K. Zhou are with the Wuhan National Laboratory for Optoelectronics (Huazhong University of Science and Technology), Key Laboratory of Information Storage System, Intelligent Cloud Storage Joint Research Center, Huazhong University of Science and Technology, Wuhan 430074, China, and also with Tencent, Shenzhen 518057, China. E-mail: {jizhang, k.zhou}@hust.edu.cn.*
- *P. Huang and X. He are with Temple University, Philadelphia, PA 19122. E-mail: {templestorager, xubin.he}@temple.edu.*
- *M. Xie, B. Cheng, Y. Ji, and Y. Wang are with Tencent Inc., Shenzhen 518057, China. E-mail: {reganxie, bencheng, raidmanji, yhwang}@tencent.com.*

Manuscript received 30 Aug. 2019; revised 13 Feb. 2020; accepted 21 Mar. 2020. Date of publication 6 Apr. 2020; date of current version 1 May 2020. (Corresponding author: Ke Zhou.) Recommended for acceptance by K. Mohror. Digital Object Identifier no. 10.1109/TPDS.2020.2985346

conservative nature of this method, i.e., it would rather miss chances to detect more disk failures than report false alarms at a higher rate.

To improve the predictive performance, several machine learning (ML) algorithms based disk failure prediction models [15], [16], [17], [18] have been proposed, which leverage training SMART data to predict disk failures. Unfortunately, these works focused on a large number of homogeneous disks which have sufficient training data. In large-scale storage system scenarios, bunches of new disks enter gradually to replace failed disks, resulting in storage systems consisting of heterogeneous disks from different vendors and different models from the same vendor as time goes on. Heterogeneous disks with numerous disk models are common in data centers [19], [20], [21]. Moreover, in evolving storage systems, some disk models are dramatically fewer than others and we call this relatively small amount of disks *minority* disks (conversely the large amount of disks as *majority* disks) in large data centers of heterogeneous disk systems. We found that about 25 percent of disks with numerous models (more than 50) are minority disks in two real-world data centers as detailed in Section 3.1. Due to the small sample and insufficient training data of minority disks, traditional ML algorithms using the training data of minority disks would dramatically increase the risk of overfitting (Section 3.1) or poor generalization [22] which will weaken the performance of predictive models and seriously affect the reliability of the storage system. Therefore, we are poised to develop a disk failure prediction model *TLDFP* to predict failures for minority disks under the condition of having rich heterogeneous disk datasets. Our basic idea is to predict minority disk failures from the available majority disk datasets, which is an application of transfer learning.

In this paper, we aim to seek answers to the following problems: *(1) What* is the definition of a minority disk dataset as far as failure predication is concerned? *(2) Why* should we use transfer learning for minority disks failure prediction? *(3) How* to use transfer learning methods to predict minority disks failure? *(4) When* to use transfer learning for minority disks failure prediction? Besides, when applied to three real-world datasets from the public *Backblaze* and *Tencent* which is one of the largest social network companies in the world, our method *TLDFP* achieves on average 96 percent failure detection rate with 0.5 percent false alarm rate based on 5 disk manufacturers (Hitachi, Seagate, Hitachi Global Storage Technologies and Western Digital) in 3 different storage media (HDD, SSD and NVMe) when making cross-disk models failure prediction in addressing realistic system challenges.

2 Background and Related Work

2.1 SMART Technology

Almost all hard disk drives, flash-based SSDs and NVMes come with built-in Self-Monitoring, Analysis and Reporting Technology, which are indicators of disk health status. The specification of SMART technology contains up to 30 attributes, reporting various disk operating conditions. SMART data directly or indirectly reflects the health condition of disks and even contains some statistical information. SMART data can be obtained through specified disk protocols upon which the disk manufacturer reached agreement. The disk would raise an alarm if the value of an SMART attribute crosses the corresponding predefined threshold. Each SMART attribute entry consists of five elements described as a tuple *(ID, Normalized, Raw, Threshold, Worst)*.

- *ID:* The designated sequence number of the SMART attribute.
- *Normalized:* Current or last normalized value (most are normalized to a value between the best value 253 and the worst value 1 calculated by manufacturer-specific algorithms using its raw value).
- *Raw:* The original value corresponding to counts or physical states provided by a sensor and vendor-specific.
- *Threshold:* The threshold value beyond which a disk alarms a failure.
- *Worst:* The lowest or worst value for a given attribute.

Not all five elements in a tuple are used. In our paper, we focus on the first three elements $(ID, Normalized, and Raw)$ in our collected datasets. For convenience, we use *"smart_ID_Raw: V"* to denote the raw value of a SMART attribute whose ID is V. For example, *smart_1_Raw: 10* means that the raw value of the read error rate attribute (ID: 1) is 10 and *smart_5_Normalized: 56* means that the normalized value of the reallocated sectors count attribute (ID: 5) is 56. More specific information about the SMART attributes we use in our evaluation is given in Table 6.

2.2 Large Scale SMART Data Collection

We not only used the publicly available SMART datasets from *Backblaze*,[1] but also collected the real-world datasets from *Tencent* Inc.,. *Tencent* Inc., founded in November 1998, is currently one of the world's biggest internet companies. In China, *Tencent* has four major data centers located in Shenzhen, Tianjin, Shanghai and Chongqing, respectively. Among them, Tianjin data center is the largest in Asia. According to *Tencent*'s statistics, as of 2019, *Tencent* hosts a total of more than 700 thousand servers and 8 million disks. In the *Tencent* Cloud Foundation Department where we obtained our SMART dataset in this paper, there are about 400 thousand servers (5 million disks) supporting a variety of business applications, such as WeChat, Qcloud (Cloud of *Tencent*), TencentVideo, and the QQ photo store QQphoto.

We configure the collection interval as one hour. In practice, the size of one sampled SMART data per disk is about 2 KB, which results in 201.4 GB data per day. Moreover, suppose the disk life is about 4-5 years, then we need to store 350.1 TB SMART data for a period of five years. In each hour, nearly 400 thousand servers send their SMART data to one server, which parses, processes and stores massive amounts of data at the same time. It imposes a great challenge to collect and store such a large-scale set of SMART data. To well handle this situation, we have proposed a scalable framework for collecting large-scale SMART data and it has been deployed in the *Tencent*'s data centers in practice.

1. https://www.backblaze.com/b2/hard-drive-test-data.html

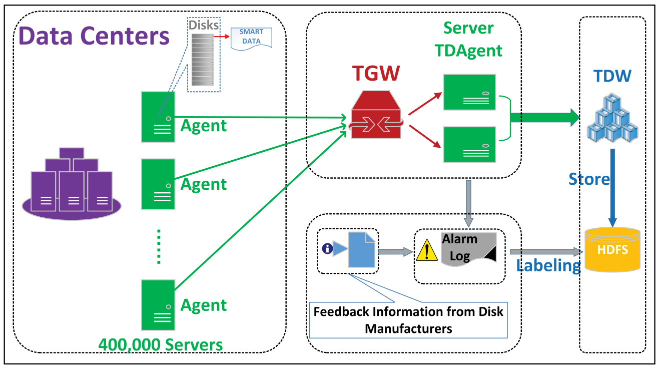


Fig. 1. The scalable SMART data collection framework deployed in the *Tencent* Inc. data center.

As it is shown in Fig. 1, the scalable SMART data collection framework mainly contains five components:

- *Agent*: An agent runs on the same server with multiple disks, typically 11 disks in *Tencent*'s servers. It periodically issues protocol-compliant requests to disks to obtain SMART data and sends SMART data files to two dedicated servers via *Tencent* Gate Way (*TGW*).
- *TGW*:2 Due to all servers in *Tencent* data centers not locating in the same network and high concurrency easily happening, we leverage the open source technology Tencent Gate Way to forward data file send requests. *TGW* enables multi-network unified accesses and also supports automatic load balancing. Upon receiving SMART data files from agents, *TGW* forwards them to one of two dedicated servers in a random manner (For illustration purpose, we draw two servers in the figure. However, *TGW* can easily scale up to connect more servers as needed).
- *TDAgent*: A dedicated server which receives raw binary SMART data files from *TGW* and converts them to regular text files for data transformation (these files will remain for 3 days temporarily). After this processing, it then periodically forwards the resultant text files to *TDW* for further storage and analysis.
- *Alarm Log*: The extracted disk alarm events and the feedback information from disk manufacturers, which are used for disk labeling. This alarm log is important for us to redeploy the collected data in new settings and to label disks in supervised learning.
- *TDW*:3 *Tencent* Distributed Data Warehouse *(TDW)*, is an open-source system of distributed data processing based on Hadoop, Hive and PostgreSQL which provides massive data storage and analysis functionality.

The framework of large-scale SMART data collection we proposed can effectively solve the problem of mass storage and difficult retrieval. Besides, these key technologies of our framework are all open source, thus more conducive to achieve and apply. Last but not the least, the high-quality SMART data is essential for a firm foundation for modeling of disk failure prediction.

2. http://wiki.open.qq.com/wiki/TGW
3. https://github.com/tonycody/tencent-tdw

2.3 Related Work

There have proposed a host of ML algorithms for disk failure prediction models based on SMART data. Hamerly and Elkan [23] employ two Bayesian methods to model disk failure based on SMART data from Quantum Inc., which consists of 1,927 good drives and 9 failed drives. They categorize the problem as anomaly detection and establish a mixture model called *NBEM*, short for Naive Bayes clusters trained using expectation-maximization and another method called naive Bayes classifier. They achieve failure detection rates of 35-40 percent for *NBEM* and 55 percent for naive Bayes classifier at about 1 percent *FAR*. Hughes *et al.* [24] explore two statistical methods to improve predictive performance. They explore the capabilities of statistical tests like the rank sum test and OP-ed single variate test, and test both methods with 7,744 drives data (out of which 36 are failures) from two different disk models spanning across a period of 3 months. They achieved a failure detection rate (*FDR*) of 60 and 0.5 percent false alarm rate (*FAR*). Murray *et al.* [25] compare the predictive performance of Support Vector Machine (SVM), rank-sum test, unsupervised clustering and reverse arrangements test.

Zhu *et al.* [15] explore the capability of a Backpropagation (BP) neural network and an improved SVM model to establish the prediction model based on SMART data. Many researchers use a SVM [26] because they claim SVM can efficiently perform a non-linear classification using the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces [27], [28]. In order to improve the stability and interpretability of the disk failure prediction model, Li *et al.* [29] propose a new hard drive failure prediction model based on Classification And Regression Trees (CART). The Regression Tree can give the disk a heath assessment rather than a simple classification result. Gradient Boosted Regression Tree (GBRT [30]) has been proposed to model disk failure [31], [32], where GBRT is a gradient descent boosting technique based on tree averaging, and is an accurate and effective ML method that can used for both regression and classification problems. To avoid over-fitting, the GBRT algorithm trains many tree stumps as week learners, rather than full, high variance trees. Moreover, the Regularized Greedy Forests (RGF [33]) approach is a powerful, non-linear classification method. It is a variation of GBRT in which the structure search and optimization are decoupled and it utilizes the concept of structured sparsity to perform greedy search directly over the forest nodes based on the forest structure. Mirela Madalina Botezatu *et al.* employ this method to model disk failure and achieve good results [34]. Xu *et al.* [35] present a Recurrent Neural Networks (RNN [36], [37]) method to leverage sequential information for predicting hard disk failure. They use a dataset collected from a real-word data center containing 3 different disk models represented as W, S and M and establish the prediction model for those disk models, respectively. They model the long-term dependent sequential SMART data and demonstrate the capability of their predictive model. More recently, Mahdisoltani *et al.* [38] propose to use traditional ML algorithms to predict disk sector errors using SMART

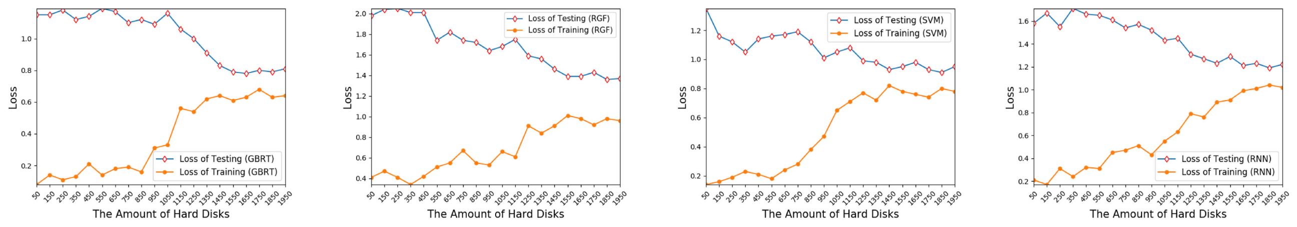


Fig. 2. The training loss and testing loss of four popular traditional ML algorithms. Note that the y-axis means loss values as the dataset size increases.

datasets. Our goals in this paper are to make whole disk failure predictions which require much higher accuracy due to cost consideration. *Note that all these studies focus only on HDDs and none of them has investigated NVMe SSDs.*

As previously mentioned, the need for transfer learning occurs when there is a limited availability of training data from a new disk model, which regularly happens to evolving storage systems. With big data repositories becoming more prevalent, using existing datasets that are related to, but not exactly the same as, a target domain of focus point or interest makes transfer learning solutions an attractive approach. There are various applications in which transfer learning has been successfully applied to, including multi-language text classification [39], [40], image classification [41], human activity classification [42], text sentiment classification [43], Web document classification [44] and so on. Unsurprisingly, in recent years, researchers have started to use transfer learning method to solve minority disks failure prediction problems [20], [34]. Mirela Madalina Botezatu *et al.* proposed the sample selection de-biasing method [34], which we denoted as *SSDB* in our paper. Its main idea is to train a classifier that can rank the observations linked to a specific disk model based on their similarity to samples pertaining to the target disk model. This method is also a single-source domain transfer learning method in spirit similar to *TLDFP* algorithm. FLF Pereira *et al.* proposed the multi-source domain transfer learning for Bayesian network [20], which we denoted as *TLBN* in our paper. It proposes a new source building method called clustering-based information source and groups several HDDs according to their similarity to build a novel information source for transfer learning. Although these methods also provide a solution to minority disks failure prediction, we have compared *TLDFP* with them and show our approach delivers better predictive performance. Besides, note that our work is the first to systematically (*What, Why, How and When*) propose using the transfer learning method to solve minority disks failure prediction based on SMART attributes for large-scale, active, evolving storage systems.

3 Preliminary Study and Motivation

In this section, we define minority disk datasets via experimental examination, investigate the distributions of SMART data, and justify *why* we use transfer learning for minority disk failure prediction.

3.1 Minority Disk Datasets

As mentioned previously, we aim to improve disk failure predictive performance for a disk dataset which has insufficient training data and where traditional ML algorithms deliver suboptimal performance. In this section, we give the definition of a *Minority Disk Dataset* and quantitatively evaluate them via experimentally showing the *training loss* and *testing loss* [45] of four popular ML algorithms (GBRT, RGF, SVM, and RNN) in disk failure prediction [28], [31], [34], [35]. A loss is a number indicating how bad a model's prediction. Note that we have added a regularization term to construct the loss function in all four methods which can effectively prevent over-fitting caused by the model having a very large number of parameters. Fig. 2 illustrates the results. As can be seen from the figure, with the dataset increasing, the loss of training set increases to a certain extent while the loss of testing set decreases because the increase in dataset leads to more complex situations where the training model needs to be fitted. More specifically, when the amount of disks is less than 1,500, the gap between the loss of training and testing decreases as the dataset enlarges, which is called *over-fitting* caused by minority disks. When the amount of disks goes over 1,500, the gap becomes smaller and stabilizes. Therefore, we can draw the following conclusions: (1) A disk dataset containing fewer than 1500 disks could lead to over-fitting which we name it as *Minority Disk Datasets*; (2) The four popular traditional ML algorithms cannot deliver satisfactory performance when the dataset contains fewer than 1,500 disks. As far as we know, we are the first to define minority disk datasets and quantitatively evaluate them through extensive data analysis and experiments. We studied two real data centers and categorized the disk quantities by the threshold of 1500. As shown in Table 1, in data center *BackBlaze*, *91* different disk models only account for less than *24 percent* of all disks while 12 models account for more than 76 percent. We call these 91 models minority disks. A similar observation has been found in data center of *Tencent*.

The above description and analysis implies that making disk failure prediction for minority disks is a realistic problem that needs to be resolved.

3.2 The Baseline Results of Using Traditional ML Only Trained on Minority Disk Datasets

In order to investigate the predictive results of using traditional ML methods *only* trained on minority disk datasets, we use 7 minority disk models from 5 disk manufacturers

TABLE 1
Characteristics of Disk Population

Data Center	Disk Number	Disk Models	Total Number	Percentage
Backblaze	$\geq$ 1500	12	114,570	76.61%
	$<$ 1500	**91**	34,978	**23.39%**
Tencent	$\geq$ 1500	8	52,235	73.32%
	$<$ 1500	**52**	18,996	**26.67%**

TABLE 2
The Results of Minority Disk Failure Prediction via Traditional ML on SATA SSDs From *STX* and *WDC* and NVMe SSDs From *SAMSUNG*

Type	Methodology	Manufacturer	FDR	FAR
HDD	GBRT	*HDS/STX/HGST/WDC*	27.3%/37.5%/31.6%/38.5%	29.0%/19.4%/17.6%/21.8%
	RGF	*HDS/STX/HGST/WDC*	36.4%/50.0%/47.4%/53.8%	44.3%/22.4%/ 53.4%/36.6%
	SVM	*HDS/STX/HGST/WDC*	50.0%/41.7%/57.9%/30.8%	20.7%/47.8%/ 24.0%/43.7%
	RNN	*HDS/STX/HGST/WDC*	40.9%/33.3%/36.8%/30.8%	28.3%/31.3%/ 39.7%/38.7%
SSD	GBRT	*STX/WDC/SAMSUNG*	23.7%/38.2%/52.8%	25.5%/27.2%/40.1%
	RGF	*STX/WDC/SAMSUNG*	35.6%/46.0%/60.4%	20.5%/32.7%/30.1%
	SVM	*STX/WDC/SAMSUNG*	15.6%/39.5%/47.2%	16.3%/41.4%/22.4%
	RNN	*STX/WDC/SAMSUNG*	40.7%/52.6%/62.3%	14.3%/22.5%/30.9%

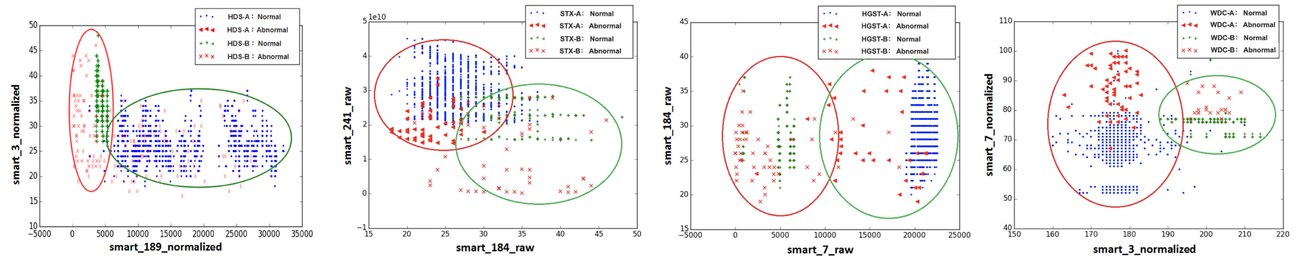
to conduct this experiment based on four popular traditional ML methods: GBRT, RGF, SVM and RNN which include the popular tree structure algorithms and deep learning algorithms and have been commonly used in disk failure prediction. Note that we only use 70 percent of the minority disk datasets as training sets and the remaining 30 percent as testing sets without any other datasets. Besides, we consistently use the following acronyms for these five vendors throughout the paper which are also used in their disk models: Hitachi (*HDS*) and Seagate (*STX*) from *Backblaze*, Hitachi Global Storage Technologies (*HGST*), Western Digital (*WDC*) and *SAMSUNG* come from *Tencent*. As can be seen from the Table 2, none of the four traditional ML methods can deliver a high *FDR* and low *FAR*. We know the poor predictive performance due to over-fitting caused by using small homogeneous datasets based on traditional ML.

We have also conducted experiments to directly use large datasets of the available majority disks to predict minority disk failures based on the four popular traditional ML techniques, the performance is not satisfactory either (details are shown in Fig. 6 in Section 5). To understand the reason, we analyze the SMART data distributions as below.

3.3 SMART Data Distributions

It is interesting to observe that the values of SMART attributes indicating disk health conditions of different disk models from the same manufacturer exhibit similar distribution patterns. We have analyzed both the publicly available SMART dataset from *Backblaze* and a dataset from the data center of *Tencent*. Fig. 3 shows the revealed SMART data distribution patterns. Each subfigure shows a pair of SMART attribute values' distribution pattern (we also investigate many other different SMART attributes, which lead to similar results) of two different disk models from the same manufacturer and each circle is used to highlight different disk models. As it is evidently shown, the relationship between *Abnormal* and *Normal* states indicated by the two SMART attributes of two disk models shows a similar pattern, with only the difference of SMART values being in different ranges. Figs. 3a, 3b, 3c and 3d respectively show that the *Abnormal* state is above, below, and to the left of the *Normal* state for the two disk models from the same manufacturer. Furthermore, the SMART values are distributed in different spectrums. Take Fig. 3b Seagate as an example, the distribution region of model *STX-B* is right-lower than that of model *STX-A*. Note that even if we ignore the y-axis (smart_241_Raw, LBAs written), there are still differences in the distribution of SMART data from these two different disk models (the distribution region of model *STX-B* is right to that of model *STX-A*). Traditional ML algorithms deliver good predictive performance only when both training and testing data are drawn from the same distribution [46]. Therefore, they fail to perform satisfactorily when it comes to cross-disk models failure prediction due to different distribution spectrums as revealed in Fig. 3.

In order to take an in-depth look at the distributions of SMART values of different disk models from the same manufacturer and further motivate the transfer learning from one disk model to a different model and explain why we use transfer learning for minority disks failure prediction, we investigate the differences in the distributions of SMART data and present a intuitive analysis comparing different disk models from the same manufacturer. Probability Density



(a) *HDS* (b) *STX* (c) *HGST* (d) *WDC*

Fig. 3. The distributions of two SMART attributes of two disk models from four manufacturers, i.e., *Hitachi*, *Seagate*, *HGST*, and *WDC*. Each subfigure shows the *Abnormal* and *Normal* states indicated by a randomly chosen pair of SMART attributes of two disk models. These four subfigures indicate that two disk models of each manufacturer exhibit similar failure patterns represented by the two SMART attributes distributions and the SMART data are distributed in different value ranges, which motivates us to apply transfer learning to make cross-model disk failure predictions.

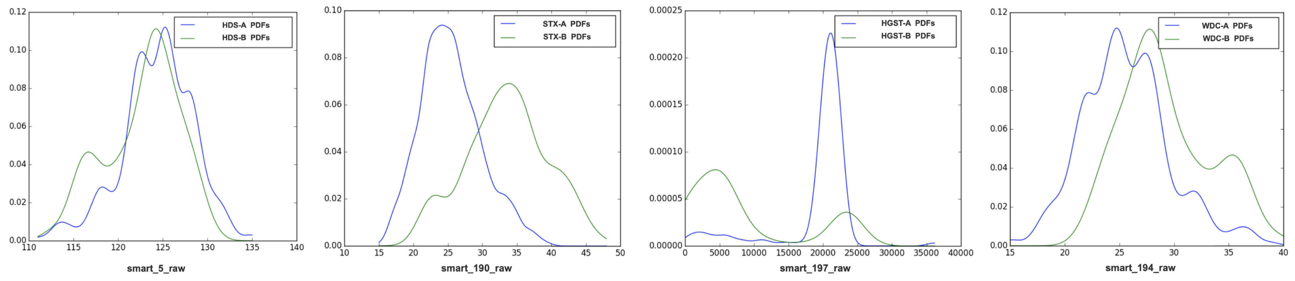
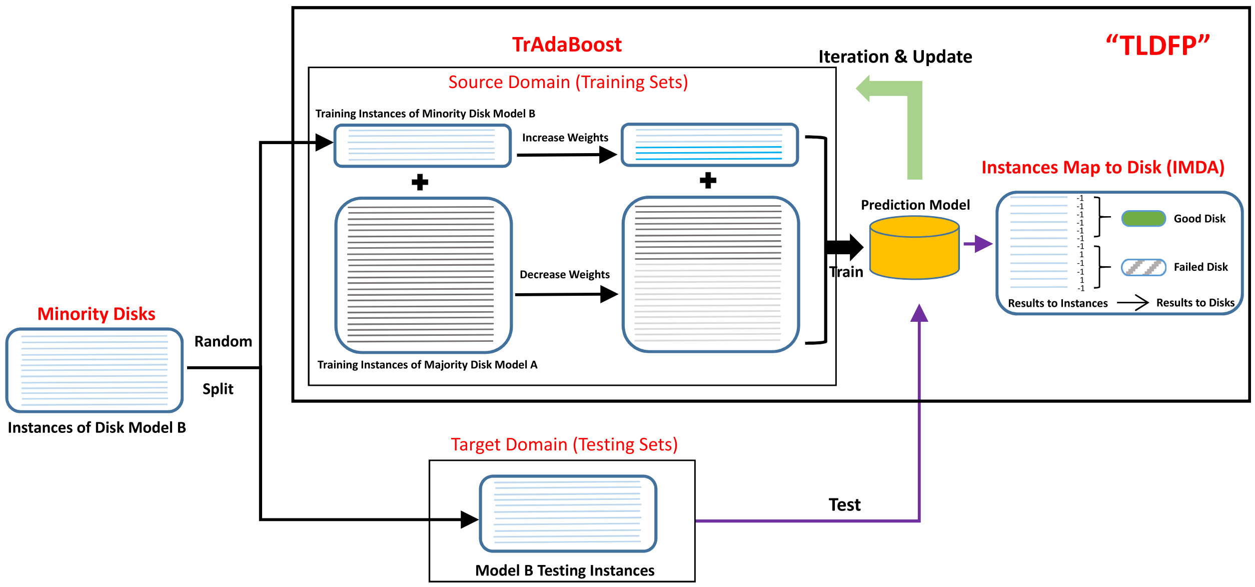
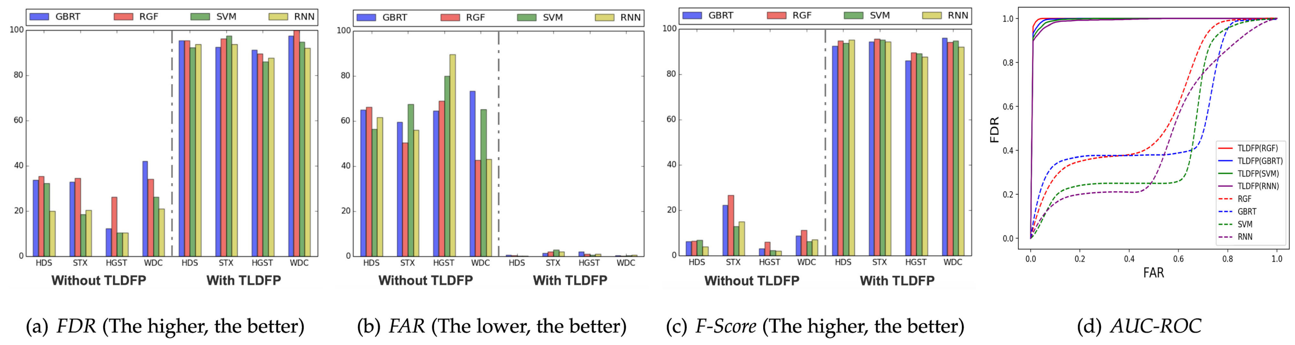


Fig. 4. PDFs of a SMART attribute value of two different disk models from four manufacturers.



TrAdaBoost
"TLDFP"
Iteration & Update
Source Domain (Training Sets)
Training Instances of Minority Disk Model B
Increase Weights
Decrease Weights
Training Instances of Majority Disk Model A
Prediction Model
Train
Instances Map to Disk (IMDA)
Good Disk
Failed Disk
Results to Instances
Results to Disks
Minority Disks
Random
Split
Instances of Disk Model B
Target Domain (Testing Sets)
Model B Testing Instances
Test

Fig. 5. The overall structure of *TLDFP*, which contains the transfer learning algorithm *TrAdaBoost* and the *Instances Map to Disk Algorithm (IMDA)*. Source domain contains a fully labeled dataset of majority disk model A and a small portion of labeled dataset of minority disk model B. The testing data in the target domain is the remaining unlabeled dataset of minority disk model B.



(a) *FDR* (The higher, the better) (b) *FAR* (The lower, the better) (c) *F-Score* (The higher, the better) (d) *AUC-ROC*

Fig. 6. The results of *FDR*, *FAR*, *F-Score* and *AUC-ROC* using four disk models based on *TLDFP* compared to four traditional ML methods.

Function statistic (PDFs) is frequently used to describe the intensity of continuous random variable. For easy observations, we use the Gaussian Kernel Density Estimation (GKDE) as the kernel function, which results in smooth curves.

Fig. 4 shows the PDFs of the value of a SMART attribute of two models from four manufactures. The distributions of the SMART data of two disk models are different but similar in that they show similar spikes though at different points and magnitudes. We refer to this phenomenon as *covariate shift* [47] among relevant predictors between different models from the same manufacturer. Therefore, we conclude that different disk models from the same manufacturer exhibit varying SMART value distributions. For the problem of minority disks failure prediction, the implication is that a prediction model built upon traditional ML algorithms using training data from one disk model is not applicable to other different models even from the same manufacturer. Therefore, to leverage a prediction model for a disk model built on adequately sufficient SMART training data to build a predictive model for a different model for which there is only limited training data, we could employ transfer learning algorithm which is inherently suitable for transferring health state information from one disk model to another disk model from the same manufacturer.

Considering the above regularity of relationship between *Abnormal* and *Normal* disk states and varying SMART data distribution spectrums across different disk models, we are motivated to apply transfer learning to predict disk failures for minority disks using the knowledge from the majority disks, which we name as *TLDFP*.

4 Minority Disk Failure Prediction

In this section, we answer the questions of *how and when* to use transfer learning for minority disk failure prediction. Specifically we detail our transfer learning method for minority disks failure prediction *TLDFP*, followed by our method of selecting source domain based on *KLD*.

4.1 *TLDFP*: Transfer Learning for Minority Disk Failure Prediction

We elaborate on *how* to use transfer learning to predict minority disk failure in this section. Fig. 5 illustrates the overall structure of our proposed predictive method *TLDFP*. It main consists of two components: a transfer learning algorithm *TrAdaBoost* [48] and an instance map to disk algorithm *IMDA*. Note that we randomly divide the SMART data of minority disk model B into two parts. The first part includes a small portion (e.g., 10 percent) of the labeled target domain data, which is then put together with the data of majority disk model A as a combined source domain to establish the relationship between the two different disk models so as to reduce variation between their distributions. The other part contains the remaining unlabeled data as testing data. Then we use our *TLDFP* method to establish a predictive model and make failure prediction for the minority disk model B training data. With the above description, the problem we aim to solve in this paper can then be formally defined as: given enough labeled training data S_a, a small amount of labeled training S_b and unlabeled testing data T_b, the main objective is to leverage the useful portions of S_a and S_b and train a classifier C which achieves a good performance of classifying the unlabeled training set T_b. The *TrAdaBoost* is an extension of the traditional ML method *AdaBoost*. *AdaBoost* is an iterative algorithm and its key procedure includes training several different weak classifiers with different weights and then consolidating those weak classifiers to a strong classifier to boost predictive performance. According to the *AdaBoost* algorithm, it first gives an initial weight to all training instances. When an instance in the source domain is found to be misclassified, we consider this instance as difficult to classify and thus increase its weight. In this way, the significance of this instance will become greater in the next iteration. However, *AdaBoost* is a traditional ML method that can only build effective predictive model for testing data which has the same distribution as the training data. In the transfer learning algorithm *TrAdaBoost*, when an instance of disk model B from the combined source domain is misclassified, we increase the weight of this instance in the next iteration, which is similar to *AdaBoost*. However, when an instance of disk model A is mispredicted, the instance is assumed to be different from disk model B. Therefore, unlike *Adaboost*, it decreases the weight of that instance in the next iteration to reduce its influences on the target domain. The details of *TrAdaBoost* are showed in Algorithm 1.

Algorithm 1. TrAdaBoost Algorithm

```
Require:
    The source domain containing labeled disk model $A$ data $S_a = \{a_1, a_2, \ldots, a_n\}$, a small amount of labeled disk model $B$ data $S_b = \{b_{n+1}, b_{n+2}, \ldots, b_{n+m}\}$, the target domain containing all unlabeled disk model B data $T_b$, and the maximum iteration $T$.
    Note: $n$ is the number of disk model $A$ data and $m$ is the number of disk model $B$ data in the source domain
1: Begin;
2: Initialize weights distribution $W_j^t$, where $t$ is the sequence of iteration, i.e., $W^1 = \{w_1^1, w_2^1, \ldots w_{n+m}^1\}$:
       $$w_j^1 = \begin{cases} 1/n, & j = 1, 2, \ldots, n \\ 1/m, & j = n, n+1, \ldots, n+m \end{cases} \quad (1)$$
3: Set $\varphi = 1/(\sqrt{2 \ln n/T} + 1)$;
4: for $t = [1, T]$ do
5:     Set the weights of instances as:
           $$I^t = \frac{W^t}{\sum_{j=1}^{j=n+m} w_j^t} \quad (2)$$
6:     Apply the basic learner to $S_a$ and $S_b$ with weights of instances $I^t$, and also to the unlabeled target domain disk model $B$ dataset $T_b$. We can achieve a classifier $h_t$;
7:     Calculate the error rate of $h_t$ on the labeled source domain disk model $B$ datasets $S_b$, note that $c(x_j)$ is the true label of the $j$th SMART sample $x_j$ :
           $$\epsilon_t = \sum_{j=n+1}^{n+m} \frac{|c(x_j) - h_t(x_j)| w_j^t}{\sum_{j=n+1}^{n+m} w_j^t} \quad (3)$$
8:     Set $\varphi_t = \epsilon_t/(1 - \epsilon_t)$. Note that if $\epsilon_t$ is greater than $1/2$, it needs to be reset to $1/2$;
9:     Update the weight distributions as below:
           $$w_j^{t+1} = \begin{cases} w_j^t \varphi^{|c(x_j) - h_t(x_j)|}, & j = 1, \ldots, n \\ w_j^t \varphi_t^{|c(x_j) - h_t(x_j)|}, & j = n, \ldots, n+m \end{cases} \quad (4)$$
10: end for
11: Map the instance to disk results using $IMDA$;
Ensure: Classify the disk as good(-1) or failed(1);
```

The input of the *TrAdaBoost* algorithm includes two disk models' training and testing data, and the maximum number of iterations T. It initializes the weights of training data and performs the iteration process. In each cycle, we use the basic learner, such as GBRT, RGF, SVM, RNN, and the weight distribution I^t to build a classifier h_b on testing data and calculate the error rate on the labeled source domain disk model B dataset S_b. Lastly, we set the new weights based on the previous iteration results and the error rate. Note that if majority disk model A instances of the source domain are misclassified, they are considered to be different from the minority disk model B. As a result, we reduce the weights of these instances in order to reduce their influences

TABLE 3
The *KLD* Values of the PDFs in Fig. 4

Source Domain	Target Domain	SMART Attribute	KLD
$HDS-A$	$HDS-B$	5_RAW	0.61
$STX-A$	$STX-B$	190_RAW	0.89
$HGST-A$	$HGST-B$	197_RAW	1.35
$WDC-A$	$WDC-B$	194_RAW	0.56

on the predictive model in the next iteration. Specifically, we multiply the instances by $w_j^t\varphi^{|c(x_j)-h_t(x_j)|}$, where φ ranges from 0 to 1 and $c(\cdot)$ is the true label of a SMART attribute. On the other hand, if the disk model B instances in the combined source domain are misclassified, we increase the weights of these instances to gain more attention in the next iteration through multiplying these instances by $w_j^t\varphi_t^{|c(x_j)-h_t(x_j)|}$, where φ_t is greater than 1. After several iterations (we will investigate the impact of this value on the predictive performance in Section 6.4), the instances of majority disk model A in the source domain that fit minority disk model B will gain greater weights and those different from disk B will have smaller weights.

Since the inputs of the failure prediction model include many SMART instances from a lot of disks at different moments, each output result indicates the prediction result for a particular instance rather than the disk health state. Therefore, we need to map the results of multiple SMART instances to the final disk state. To achieve that, we propose an *Instances Map to Disk Algorithm* (*IMDA*) by extensive experiments and analysis using the majority disk dataset in the training progress. *IMDA* determines the final health state of a minority disk in testing progress (practical use). Specifically, it performs a prediction for each minority disk once every day, if any instance of one disk is predicted as a failure, the corresponding disk will be considered as failure. Besides, we discuss other alternative options are discussed in Section 6.5 and our method outperforms all others in terms of predictive result.

4.2 Source Domain Selection Based on KLD

When can we use *TLDFP* for minority disks failure prediction? To answer this question, we use Kullback Leibler Divergence (*KLD*), which is a metric measuring the divergence degree of one probability distribution from another expected probability distribution [49]. *KLD* values indicate the disparities between two random variable distributions. A zero *KLD* value means that the two random distributions are the same, while the *KLD* value increases as the differences between two random distributions widen. In general, the bigger a *KLD* value is, the greater differences between two distributions will be and the more difficult the knowledge transfer between two distributions will be. Table 3 gives *KLD* values corresponding the PDFs showed in Fig. 4. Table 3 shows that all *KLD* values are not equal to zeros, confirming that the respective SMART data distributions are indeed not the same. Note that as indicated in Table 3, the *KLD* value trend, which is consistent with the PDF differences increase from *HDS*, to *STX*, to *HGST*, to *WDC* shown in Fig. 4. Considering that *TLDFP* is a method to decrease the data distribution differences between source domain and target domain, so we infer that the bigger *KLD* value between one disk model and another is, the harder *TLDFP* can transfer experience. The predictive results in Section 5.2 proves our conjecture and we also conduct detailed experiments and discussions in Section 6.1. Therefore, the value of *KLD* can guide us to select appropriate majority disk dataset (source domain) for training the minority disk failure predictive model. As far as we know, we are the first attempt to present a novel method based on *KLD* values as an effective indicator to select proper majority disk models and improve disk failure prediction. Note that we also use the Jensen-Shannon Divergence (*JSD*) and Wasserstein Distance (*WD*) but find *KLD* is the best metric to guide us to select an appropriate majority disk dataset and results in better prediction performance. More specifically, *JSD* (*WD*) only achieves on average 79.6 percent (83.1 percent) *FDR* and 3.2 percent (2.1 percent) *FAR*. Our evaluation results in Section 6.1 demonstrate that our approach of using *KLD* is very effective and practical.

5 Experimental Evaluation

In this section, we evaluate the predictive performance of *TLDFP*. We first describe the methodology, followed by the experimental results of comparing *TLDFP* against four ML algorithms and two state-of-the-art transfer learning methods according to the evaluation metrics.

5.1 Methodology

We describe the characteristics of three real-world SMART datasets in our experiments and SMART attributes selection. Then we introduce four evaluation metrics commonly used in ML and some testing methods we use to conduct all our experiments.

5.1.1 Datasets and Attribute Selections

Datasets. Defining a failed disk is a difficult task in varying deployment scenarios. Based on experience and post-mortem analysis in Tencent Inc., we define a disk that cannot function properly as failed if it was replaced as part for repair. Specifically, it includes three cases: the disk has a write operation error, the system loses connection to the disk and an operation (i.e., disk scrubbing, read and write calls) exceeds the timeout threshold. Note that not all samples of failed disks need to be used in the training set; otherwise, those good samples of failed disks which are far from the actual failure would disturb the training of the detection model. Therefore, only the last 14 continuous samples (our goal is to predict disk failure 14 days in advance) before the moment of failure of the training disks can be regarded as failed samples. We use three SMART datasets from real-world data centers for evaluations. Table 4 gives the overall characteristics of the two datasets. Every disk is classified either as *"Good"* or *"Failed"*. "Sample" indicates SMART records. Each good disk or failed disk has many SMART records. As the original dataset has more samples of good disks than failure disks, we use majority class undersampling to improve training in the case of imbalanced classes to create training datasets. We have chosen a 1:3 ratio of failure disk to good disk [38]. When performing the training in traditional ML method, we divide the data into *70 percent*

TABLE 4
SMART Datasets

Data center	Duration	Good	Good Sample	Failed	Failure Sample
BackBlaze	50 months	141,891	106,867,099	7,657	7,689
Tencent	26 months	68,436	774,994,430	2795	31,574,341

training and 30 percent testing data for our experiments which is in line with existing work [20]. Note that all the results of our experiments are obtained by *cross-validation* [50] in order to avoid fortuitous accident which is common used in ML. Table 5 lists our chosen disks for evaluations.

SMART Attribute Selections. Each SMART observation can contain up to 30 meaningful SMART attributes. However, some attributes are irrelevant to our disk failure predictive model because they are immutable or have not experienced noticeable abnormal changes. Therefore, we selectively keep those attributes that are relevant to the disk health state according to a feature selection process based on Principle Component Analysis (PCA) while ignoring other irrelevant attributes. The selected SMART attributes of HDDs, SATA SSDs and NVMe SSDs are listed in Table 6. For each SMART sample, we use the *Normalized Value* and *Raw Value*. The normalized value typically represents the current value of an attribute. However, certain normalized values lose accuracy when transformed from the raw value and some raw values are more sensitive to the predictive model. We also use other methods of feature selection like [20], [34], [38]. However, the impact on the experimental results is not significant, so we do not discuss further due to limited space.

In addition, different SMART attributes have different output ranges, which will lead to different impacts on the predictive model. In order to make a fair comparison among different SMART attributes in our disk failure predictive model, we normalize the range of all selected SMART attributes using the min-max scaling which is in line with existing work [38]

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}},$$

where x is the original value of a SMART attribute, x_{max} and x_{min} are the maximum and minimum value of the attribute in the training set, respectively.

5.1.2 Evaluation Metrics

Confusion matrix [51] is a tool of visualization for interpreting the performance of machine learning algorithms.

TABLE 5
Selected Disk Models Used in Evaluations

Data center	Manufacturer	Disk model	Good	Bad
Backblaze	Hitachi	HDS722020ALA330	4774	225
		HDS723030ALA640	1048	72
	STX	ST4000DM000	37006	3157
		ST4000DX000	222	81
Tencent	HGST	HGST-A	13367	451
		HGST-B	679	63
	WDC	WDC-A	6847	259
		WDC-B	472	42
Tencent	STX (SATA SSD)	SSD-S-A	8672	397
		SSD-S-B	922	135
	WDC (SATA SSD)	SSD-W-A	13972	489
		SSD-W-B	679	76
	SAMSUNG (NVMe SSD)	NVMe-A	2653	136
		NVMe-B	392	53

TABLE 6
SMART Attributes of HDD, SATA SSD, and NVMe SSD Selected for Our Evaluations

Type	#ID	SMART Attribute Name	Attribute type
HDD	001	Raw Read Error Rate	Normalized&Raw
	003	Spin-Up Time	Normalized
	005	Reallocated Sectors Count	Normalized&Raw
	007	Seek Error Rate	Normalized&Raw
	009	Power-On Hours	Normalized&Raw
	184	I/O Error Detection and Correction	Normalized&Raw
	187	Reported Uncorrectable Errors	Normalized&Raw
	188	Command Timeout	Raw
	189	High Fly Writes	Normalized&Raw
	190	Airflow Temperature	Normalized&Raw
	193	Load/Unload Cycle Count	Normalized&Raw
	194	Temperature	Normalized&Raw
	197	Current Pending Sector Count	Normalized&Raw
	240	Head Flying Hours	Raw
	198	Offline Uncorrectable Sector Count	Normalized&Raw
	241	Total LBAs Written	Raw
	242	Total LBAs Read	Raw
SATA SSD	001	Raw Read Error Rate	Normalized&Raw
	005	Retired Block Count	Normalized
	009	Power On Hours Count	Normalized&Raw
	012	Power Cycle Count	Normalized
	171	Program Fail Count	Normalized&Raw
	172	Erase Fail Count	Normalized&Raw
	174	Unexpected Power Loss Count	Normalized&Raw
	177	Wear-Range Data	Normalized&Raw
	187	Reported Uncorrectable Errors	Normalized&Raw
	188	Command Timeout	Normalized
	195	On the Fly Reported Uncorrectable Error Count	Raw
	196	Reallocated Event Count	Normalized&Raw
	197	Read Failure block Count	Normalized&Raw
	206	Write Error Rate	Normalized
	208	Erase Count Average	Normalized&Raw
NVMe SSD	001	Temperature	Normalized&Raw
	002	Available Spare	Normalized
	003	Available Spare Threshold	Normalized
	004	Percentage Used	Normalized
	005	Controller Busy Time	Normalized&Raw
	006	Power Cycles	Normalized&Raw
	007	Power On Hours	Normalized&Raw
	008	Unsafe Shutdowns	Normalized&Raw
	009	Media and Data Integrity Errors	Normalized&Raw
	010	Error Information Log Entries	Normalized&Raw

Each column in a confusion matrix denotes the classified true class and each row denotes the predictive class. Table 7 shows the confusion matrix used in disk failure prediction. We refer to a failed disk sample as a *Positive* instance (denoted as *"P"*), and a good disk sample as *Negative* instance (denoted as *"N"*). The prediction result takes on only two values: *True* (denoted as *"T"*) or *False* (denoted as *"F"*). Therefore, *"TP"* standing for *"True Positive"*, means that a failed disk is correctly predicted and *"FP"* standing for *"False Positive"*, means that a good disk is falsely predicted as a failed disk. By a similar reasoning, we can get the meanings of both *"FN"* and *"TN"*. Using the statistics information about the four metrics, we can construct many evaluation criteria which are often utilized in judging machine learning algorithms. In our evaluations, we use the following four metrics to report the results in our experiments which are commonly used for evaluating the capability of a classification model in ML [52].

TABLE 7
Confusion Matrix Used in Disk Failure Prediction

	True failed disk	True good disk
Predictive failed disk	*TP*	*FP*
Predictive good disk	*FN*	*TN*

FDR. Failure Detection Rate ($FDR = \frac{TP}{TP+FN}$) also called *recall rate*. It captures the proportion of true failed disks that are correctly predicted as failed. The higher the *FDR* is, the better the model is.

FAR. False Alarm Rate ($FAR = \frac{FP}{FP+TN}$), the proportion of good disks that are falsely predicted as failed. The lower the *FAR* is, the better the model is.

F-Score. F-Score is a balance between *FDR* and *Prediction Precision* ($PP = \frac{TP}{TP+FP}$). *PP* is the proportion of predictive failed disks that are correctly predicted as failed. Therefore, the specific calculation formula of *F-Score* is $\frac{2*FDR*PP}{FDR+PP}$. The higher the *F-Score* is, the better the model is.

AUC-ROC Curve. The *Area Under the Curve-Receiver Operating Characteristic* (*AUC-ROC*) curve is a performance measurement for classification problem at various threshold settings. *ROC* is a probability curve and *AUC* represents degree or measure of separability. It is plotted with *FDR* against the *FAR* where *FDR* is on y-axis and *FAR* is on the x-axis. In disk failure prediction, a higher the *AUC* means the model is better at distinguishing failed and good disks.

5.1.3 Testing Methods and Configurations

To verify the effectiveness of our proposed *TLDFP*, we conduct experiments in three scenarios: 1) to use traditional ML methods only trained on the minority disk datasets, 2) to compare *TLDFP* with traditional ML techniques (*baseline*), and 3) to compare *TLDFP* with other transfer learning approaches. The settings are described below.

1) Traditional ML methods only trained on minority disk:

The detailed description see Section 3.2.

2) TLDFP with traditional ML techniques:

In this scenario, we conduct experiments to investigate the performance of minority disks failure prediction based on four traditional ML algorithms using large heterogeneous datasets for training from both different disk models and the same disk manufacturer. Table 8 shows the training and testing datasets. Note that we randomly choose 10 percent testing datasets for training with all training datasets.

3)TLDFP with other transfer learning approaches:

In addition to traditional ML techniques, we also compare our *TLDFP* with two state-of-the-art transfer learning methods (*SSDB* and *TLBN*) of predicting minority disks failure. Note that we use the same datasets for the two methods in [34] and [20] for fair comparison in all our experiments.

5.2 Experimental Results

In this section, we show the HDD, SATA SSD and NVMe SSD results of the *TLDFP* compared to traditional ML methods and other transfer learning methods with four evaluation metrics mentioned in Section 5.1.2 respectively. *Note that we had showed the poor baseline results of using traditional ML methods only trained on the minority disk datasets in Section 3.2.*

5.2.1 Evaluations Compared to Traditional ML Approaches

- *FDR/Recall Rate:* We conduct experiments to investigate the *FDR* of *TLDFP* and four popular traditional ML methods using four HDD models from two real data centers. As can be seen from the Fig. 6a, none of the four traditional ML methods can deliver a high *FDR* using large heterogeneous datasets. However, the *TLDFP* use the above GBRT, RGF, SVM, RNN algorithms respectively as the basic learners all achieved higher *FDR*.
- *FAR:* Note that the goal of our *TLDFP* is not only to achieve high *FDR* but also low *FAR* for minority disk failure prediction. The results of *FAR* are showed in Fig. 6b. All other methodologies show higher *FAR* which is unacceptable in realistic data centers. Further, none of the four traditional ML methods can deliver both a high *FDR* and a low *FAR* on minority disks except for *TLDFP*. Based on the analysis in Section 3.3, we know the poor predictive performance caused by the traditional ML methods do not have the ability to reduce the distribution difference between the minority disk datasets in target domain and majority disk datasets in source domain.

TABLE 8
Datasets of Minority Disk Failure Prediction Using Large Heterogeneous Dataset Based on Traditional ML

Data Center	Manufacturer	Training Disk Model	Testing Disk Model	Training set	Testing set
Backblaze	*Hitachi*	*HDS-A*	*HDS-B*	4774 good *HDS-A* and 225 failed *HDS-A* 105 good *HDS-B* and 7 failed *HDS-B*	943 good *HDS-B* and 65 failed *HDS-B*
	STX	*STX-A*	*STX-B*	37006 good *STX-A* and 3157 failed *STX-A* 22 good *STX-B* and 8 failed *STX-B*	200 good *STX-B* and 73 failed *STX-B*
Tencent	*HGST*	*HGST-A*	*HGST-B*	13367 good *HGST-A* and 451 failed *HGST-A* 68 good *HGST-B* and 6 failed *HGST-B*	611 good *HGST-B* and 57 failed *HGST-B*
	WDC	*WDC-A*	*WDC-B*	6847 good *WDC-A* and 259 failed *WDC-A* 47 good *WDC-B* and 4 failed *WDC-B*	425 good *WDC-B* and 38 failed *WDC-B*
	STX	*SSD-S-A*	*SSD-S-B*	8672 good *SSD-S-A* and 397 failed *SSD-S-A* 92 good *SSD-S-B* and 14 failed *SSD-S-B*	830 good *SSD-S-B* and 121 failed *SSD-S-B*
	WDC	*SSD-W-A*	*SSD-W-B*	13972 good *SSD-W-A* and 489 failed *SSD-W-A* 68 good *SSD-W-B* and 8 failed *SSD-W-B*	611 good *SSD-W-B* and 68 failed *SSD-W-B*
	SAMSUNG	*NVMe-A*	*NVMe-B*	2653 good *NVMe-A* and 136 failed *NVMe-A* 39 good *NVMe-B* and 5 failed *NVMe-B*	353 good *NVMe-B* and 48 failed *NVMe-B*

TABLE 9
The *FDR*, *FAR*, *F-Score*, and *AUC* of the Comparisons Between *TLDFP* and *SSDB*, *TLBN*

Methods	Manufacturer	*FDR*	*FAR*	*F-Score*	*AUC*
TLDFP (RGF) VS SSDB	*STX*	94.9%/ 85.8%	1.6%/ 3.0%	92.6%/ 83.7%	0.93/ 0.86
	Hitachi	97.1%/ 70.8%	0.9%/ 5.4%	95.7%/ 69.0%	0.96/ 0.81
TLDFP (RGF) VS TLBN	*STX*	91.3%/ 73.1%	0.6%/ 2.6%	91.3%/ 70.4%	0.91/ 0.83

TABLE 10
KLD Values Between Each Disk Model in the Training Sets and the Minority Disk Model in the Testing Set Using Method *TLBN*

Training Disk Model	Testing Disk Model	SMART Attribute	*KLD*
ST320005XXXX	ST33000651AS	5_RAW	5.6
ST32000542AS		190_RAW	2.7
ST1500DL003		188_RAW	7.1
ST31500341AS		190_RAW	1.5
ST31500541AS		197_RAW	0.83

- *F-Score:* Fig. 6c compares the *F-Score* of the different prediction models on the two datasets using four disk models. As can be seen, *TLDFP* has much higher *F-Score* than other different traditional ML methods. As an example, the *F-Score* of *TLDFP* with RBF as its basic learner *TLDFP(RBF)* is almost 9 times as high as the algorithm RBF for *WDC* disks from data center of *Tencent*. As we recall in Section 4.2, the bigger *KLD* value of HGST dataset will lead to more difficult knowledge transfer. This conclusion is also confirmed by the *F-Score* observations given that *the F-Score of TLDFP for HGST are generally lower that other cases*. We will further discuss this issue in detail in Section 6.1.
- *AUC-ROC Curve:* We plot the *AUC-ROC* curve in Fig. 6d using *WDC* disk model in *Tencent*. As it is shown, the *AUC-ROC* curve of *TLDFP* are all close to the top left corner and *TLDFP(RGF)* achieving the higher *AUC* value compared to other two transfer learning methods. The four traditional ML methods achieved lower *AUC* values, reflecting their poor classification ability in performing cross-disk model failure predictions.

5.2.2 Evaluations Compared to Other Transfer Learning Approaches

As it is shown in Table 9, *TLDFP* shows higher *FDR*/*F-Score* and lower *FAR* than *SSDB* and *TLBN*. The reason is that although *SSDB* matches the distribution of the source domain with the target domain, it only ranks the observations in source domain, while *TLDFP* makes more effective weight adjustments to every observation. Considering the *TLBN* is a multi-source domain transfer learning method and *TLDFP* is a single-source domain transfer learning method, we analyze the *KLD* values between each disk model in the source domain and the minority disk model in the target domain. The results are showed in Table 10. We find the data in the disk model with a large *KLD* value in the source domain (such as ST320005XXXX and ST1500DL003). However, *TLDFP* only uses the disk model which has the smallest *KLD* value as source domain. A large *KLD* value leads to difficulties for *TLBN* in mitigating the distribution differences between the source and target domains. This result also shows that single-source domain transfer learning performs better than multi-source domain transfer learning and there is a good metric (e.g, *KLD*) for evaluating differences between different domains.

5.2.3 Evaluations on SATA SSD and NVMe SSD

In addition to the experimental results of HDDs failure prediction, we also conduct experiments on SATA SSD and NVMe SSD in the three scenarios mentioned in Section 5.1.3 and verify the good performance of our *TLDFP*. Table 2 illustrates the poor results of first scenarios (see Section 3.2). As can be seen from the Table 11, whether with traditional ML methods or other transfer learning approaches, our *TLDFP* can achieve higher *FDR*, *F-Score*, *AUC* and lower *FAR* at same time. That shows the effectiveness of our *TLDFP* on different storage media.

In summary, the results have demonstrated that *TLDFP* can effectively solve the problem of minority disk failure prediction with much better predictive performance than traditional ML methods and two other transfer learning methods for the same datasets. More specifically, *TLDFP* not only delivers high *FDR*, *F-Score and AUC*, but also shows a rather low *FAR* at the same time. The main reason for the comparison results is that our *TLDFP* algorithm is able to utilize the small number of labeled target disk data to establish the relationship between source and target disk models, which helps the large heterogeneous disk model to be well trained toward the characteristics of the minority target disk model. In other words, *TLDFP* reduces the difference in data distribution between the source and target domain, which we will further discuss in Section 6.2. *Note that we don't include the results of all methods or disk models due to space limit. From all our tests, TLDFP demonstrates the best performance.*

6 Observations and Sensitivity Study

In this section, we provide several additional sensitivity studies from six aspects.

6.1 The Impact of KLD in Source Domain Selection

In order to verify our conjecture in Section 4.2 and further explain the results in Section 5.2.1 and Section 5.2.2, we analyze the relationship between *KLD* and *F-Score* in *TLDFP*

TABLE 11
The Results of Evaluations on SATA SSD and NVMe SSD

Model	Methodology	*FDR(%)*	*FAR(%)*	*F-Score(%)*	*AUC*
SSD-S-B	*TLDFP(GBRT)*	**92.6**	**0.8**	**93.6**	**0.88**
	GBRT/SSDB/TLBN	18.5/68.9/ 65.2	34.2/12.3/ 13.6	10.6/54.5/ 50.6	0.39/0.71/ 0.68
SSD-W-B	*TLDFP(RGF)*	**93.4**	**1.2**	**91.6**	**0.90**
	RGF/SSDB/TLBN	28.9/76.3/ 82.9	41.4/13.5/ 11.8	11.7/56.2/ 57.6	0.42/0.73/ 0.75
NVMe-B	*TLDFP(RNN)*	**94.3**	**1.0**	**93.4**	**0.91**
	RNN/SSDB/TLBN	50.9/66.0/ 77.4	46.4/20.2/ 16.8	20.6/41.9/ 51.2	0.51/0.58/ 0.6

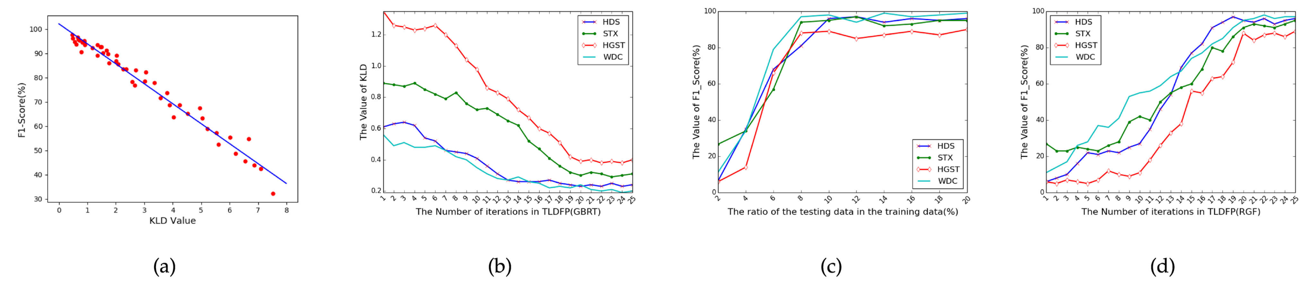


Fig. 7. (a) The fitting curves of *KLD* and *F-Score*. (b) *KLD* value decrease in training. (c) The relationship between FDR and percentage of target domain data drawn into source domain. (d) The relationship between *FDR* and the number of iterations in the *TLDFP*.

using several minority disk models as testing disk model and large heterogeneous datasets of one disk model for training from the same manufacturer in two real data centers. The results are shown in Fig. 7a and Table 12. We observed that the value of *F-Score* keeps rising as the value of *KLD* decreases. In other words, it shows that the smaller the difference in SMART data distribution between the source and target domain, the easier knowledge transfer in *TLDFP* can be. Therefore, we use *KLD* as an effective indicator for source domain selection of large heterogeneous dataset and usually select the one which has the smallest *KLD* value.

6.2 The Change of KLD Value in TLDFP

In order to more intuitively observe how the *TLDFP* method reduces the difference in data distribution between the source domain and the target domain, we record the value of *KLD* between the training set and the dataset after each iteration of the model during the training process of *TLDFP(GBRT)*, as illustrated in Fig. 7b. We can see that as the number of iterations of the model increases, the *KLD* value between the source domain and the target domain decreases continuously, and stabilizes at a smaller value when the number of iterations is about 22. This shows that our *TLDFP* model continuously reduces the difference of the SMART data distribution between the two domains in the training process, enabling us to use the large heterogeneous disk data to predict the minority disk data and realize knowledge transfer.

6.3 Varying Samples from Target Domain

As discussed previously our prediction model *TLDFP* uses a portion of labeled dataset from target domain as part of its source domain dataset. In this section, we investigate how the percentage number affects predictive performance of *TLDFP*. We report the results of *TLDFP* with RGF as its basic learner and using the disk data of *WDC* model from *Tencent* data center, as the other three basic learners show similar results. Fig. 7c shows the relationship between the *FDR* and the percentage of target domain data put in the source domain. As it clearly shows, the *FDR* increases dramatically when the percentage increases from 2 to 10 percent. When the percentage goes beyond 10 percent, the *FDR* does not continue to increase but remains a relatively high level, meaning putting more target domain data to the source domain does not help further improving predictive performance. Consequently, we randomly choose 10 percent target domain data in our previous experiments.

TABLE 12
The *F-Score* Varies With the Value of *KLD*

Data Center	Method	Training Model	Testing Model	*KLD*	*F-Score*
BackBlaze	*TLDFP* (GBRT)	*HGST-K*	*HGST-L*	2.67	76.8%
			HGST-M	1.76↓	85.9%↑
			HGST-N	0.91↓	93.5%↑
			HGST-O	0.69↓	95.7%↑
			HGST-P	0.47↓	97.6%↑
Tencent	*TLDFP* (SVM)	*STX-K*	*STX-L*	3.57	71.6%
			STX-M	2.58↓	78.2%↑
			STX-N	2.26↓	83.4%↑
			STX-O	1.35↓	93.1%↑
			STX-P	1.17↓	92.2%↓
			STX-Q	0.71↓	95.3%↑
			STX-R	0.66↓	96.6%↑

6.4 The Impact of Iterations

The *TrAdaBoost* algorithm takes an input parameter to cap the iterations performed by the algorithm. In this section, we study how the iteration parameter affects predictive performance in terms of *F-Score*. We report the results of *TLDFP* with RGF as its basic learner. Fig. 7d shows how *F-Score* changes as the number of iterations varies for different datasets. From this figure, we can make similar observations as in the preceding subsection. As the number of iterations increases, the *F-Score* increases quickly and reaches a stable level at 22 iterations (fast convergence). It also shows that as the algorithm adjusts instance weights in each iteration, *TLDFP* gradually adjusts to converge toward the testing data. Since the number of 22 represents a inflection point, we adopt this iteration number in our previous experiments. Note that the number of iterations is not fixed according to the different datasets and model parameters. In general, the model would be stopped for training after the model loss bottoms.

6.5 The Sensitivity Study of IMDA

In Section 4.1, we introduced the algorithm *KLD*. Specifically, if any instance is classified as failure, the corresponding disk is considered as failure. Here we conduct experiments to evaluate the impact of different instances, specifically, 1 instance, 1/3 of all instances, 1/2 of all instances, 2/3 of all instances, and all instances being classified as failure.

The result of this experiment using *TLDFP(RNN)* based on the *WDC* disk model data in *Tencent* is showed in Table 13. It is clear that the option we use (one instance failure indicates the corresponding disk failure) achieves the

TABLE 13
The Sensitivity Study Results of *IMDA* in *TLDFP*

Methods	Metrics	1	1/3	1/2	2/3	All
TLDFP	*FDR*	95%	32%	24%	8%	3%
	FAR	0.7%	0.7%	0.5%	0.2%	0.2%

best performance. In practice, we perform a prediction for each minority disk once a day. If any instance of the disk is predicted as a failure, the corresponding disk will be considered as failure.

6.6 Cost Benefits

As we have demonstrated so far that *TLDFP* achieves both high *FDR* and low *FAR*. It is easily understandable that improving *FDR*, i.e., disk failures are correctly predicted, can reduce the probability of data loss occurring. In this section, we discuss the benefits brought by low *FAR* in ML methods. A *FAR* will trigger responsive procedures to be launched, e.g., data migration, disk replacement, etc., which incurs extra cost to the IT management. Assume that disk failures are independent events with a probability of ϵ_d and the number of disks in a data center is N. A disk will fail after a certain period of time, which is called disk lifespan. For instance, the *Tencent*'s data centers assume the lifespan T to be 4-5 years. The disk failure rate over a period of time is dependent on the time T and ϵ_d. Therefore, we use $f(T, \epsilon_d)$ to denote the failure rate of disks in a data center that are failed over a period of T. The total number of failed disks in the period T can be expressed as N*$f(T, \epsilon_d)$ and the total number of failed disks per unit time can be described as:

$$\frac{N * f(T, \epsilon_d)}{T}.$$

The total number of good disks is thus given by

$$N * (1 - \frac{f(T, \epsilon_d)}{T}).$$

Given a *FAR*, the number of all non-faulty disks which are wrongly predicted to be failed can be described as

$$FAR * N * (1 - \frac{f(T, \epsilon_d)}{T}).$$

Assume $c(\$)$ is the statistic average cost of replacing one disk (including many disks out of the warranty period), which contains the cost of manual replacement c_1 and the cost of a new disk c_2 as well as the cost c_3 caused by copying the data from the failed disk to a new disk. The total additional cost is

$$Cost = c * FAR * N * (1 - \frac{f(T, \epsilon_d)}{T}).$$

Therefore, if *FAR* = 0, the cost will be 0. On the other hand, if *FAR* = 1, meaning all good disks are wrongly predicted to be failed and replaced, the cost would be maximum. Last but not the least, the total additional cost caused to the data center is proportional to the *FAR* of the prediction model. The larger *FAR* is, the more additional cost will be. The traditional machine learning methods and other transfer learning methods, lead to an average of 60 and 4 percent *FAR*, when they are used to predict disk failures for minority disks, while our *TLDFP* achieves an average of 0.5 percent *FAR*, which can be translated to 120X and 8X reduction in additional cost. According to an average cost was estimated as c = \$426 in *Tencent* data centers in 2018, the *Tencent* company can save about \$105 to \$1,800 milions per year. In addition, we get a higher *FDR* with lower *FAR* which means *TLDFP* predicts true failed disks as a good disk rarely, reducing the inestimable cost due to disk failures.

7 Conclusion

In this paper, we develop a model called *TLDFP* to effectively predict minority disk failure leveraging the transfer learning based on different storage media, where traditional ML approaches perform poorly. Our main contributions include: (1) we are the first to define minority disk datasets and quantitatively evaluate them through extensive data analysis and experiments in data centers of heterogeneous disk models, (2) we are the first to present a novel method based on *KLD* values to select proper majority disk models, and (3) we develop a method of making crossing-disk model (HDD, SATA SSD and NVMe SSD) failure prediction, which has important practical applicability as different disk models are gradually placed into the realistic storage systems to replace failed disks. Our experiments with three datasets from real-world data centers have shown that *TLDFP* well outperforms representative traditional ML methods and existing transfer learning approaches in terms of failure detection rate and false alarm rate. *TLDFP* achieves on average 96 percent failure detection rate with only 0.5 percent false alarm rate in performing crossing-disk models failure prediction and reduces inestimable cost to data centers.

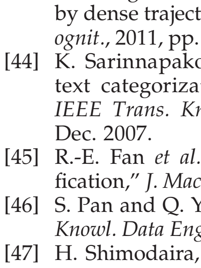
Acknowledgments

This work was supported by the Innovation Group Project of the National Natural Science Foundation of China No.61821003, the National Key Research and Development Program of China under Grant No.2016YFB0800402 and the U.S. National Science Foundation Grants CCF-1717660 and CCF-1813081. A preliminary version [1] appeared in the proceedings of the 48th International Conference on Parallel Processing (ICPP), 2019.

References

[1] Z. Ji *et al.*, "Transfer learning based failure prediction for minority disks in large data centers of heterogeneous disk systems," in *Proc. 48th Int. Conf. Parallel Process.*, 2019, pp. 1–10.
[2] B. Schroeder and G. A. Gibson, "Disk failures in the real world: What does an MTTF of 1, 000, 000 hours mean to you?" in *Proc. 5th USENIX Conf. File Storage Technol.*, 2007, pp. 1–16.
[3] E. Pinheiro *et al.*, "Failure trends in a large disk drive population," in *Proc. 5th USENIX Conf. File Storage Technol.*, 2007, pp. 17–28.
[4] L. N. Bairavasundaram *et al.*, "An analysis of latent sector errors in disk drives," in *Proc. ACM SIGMETRICS Int. Conf. Meas. Model. Comput. Syst.*, 2007, pp. 289–300.
[5] J. Meza *et al.*, "A large-scale study of flash memory failures in the field," in *Proc. ACM SIGMETRICS Int. Conf. Meas. Model. Comput. Syst.*, 2015, pp. 177–190.

[6] B. Schroeder *et al.*, "Flash reliability in production: The expected and the unexpected," in *Proc. 14th USENIX Conf. File Storage Technol.*, 2016, pp. 67–80.
[7] B. Calder *et al.*, "Windows azure storage: A highly available cloud storage service with strong consistency," in *Proc. 23rd ACM Symp. Operating Syst. Princ.*, 2011, pp. 143–157.
[8] C. Huang *et al.*, "Erasure coding in windows azure storage," in *Proc. USENIX Conf. Annu. Tech. Conf.*, 2012, pp. 15–26.
[9] S. Huang and S. Fu, Q. Zhang, and W. Shi, "Characterizing disk failures with quantified disk degradation signatures: An early experience," in *Proc. IEEE Int. Symp. Workload Characterization*, 2015, pp. 150–159.
[10] Y. Xu *et al.*, "Improving service availability of cloud systems by predicting disk error," in *Proc. USENIX Conf. Usenix Annu. Tech. Conf.*, 2018, pp. 481–494.
[11] H. S. Gunawi *et al.*, "Fail-slow at scale: Evidence of hardware performance faults in large production systems," *ACM Trans. Storage*, vol. 14, no. 3, pp. 23:1–23:26, 2018.
[12] D. A. Patterson *et al.*, "A case for redundant arrays of inexpensive disks (RAID)," in *Proc. ACM Int. Conf. Manage. Data*, 1988, pp. 109–116.
[13] B. Allen, "Monitoring hard disks with SMART," *Linux J.*, vol. 117, pp. 60–65, 2004.
[14] J. F. Murray *et al.*, "Machine learning methods for predicting failures in hard drives: A multiple-instance application," *J. Mach. Learn. Res.*, vol. 6, pp. 783–816, 2005.
[15] B. Zhu, G. Wang, X. Liu, D. Hu, S. Lin, and J. Ma, "Proactive drive failure prediction for large scale storage systems," in *Proc. IEEE 29th Symp. Mass Storage Syst. Technol.*, 2013, pp. 1–5.
[16] W. Yang, D. Hu, Y. Liu, S. Wang, and T. Jiang, "Hard drive failure prediction using big data," in *Proc. IEEE 34th Symp. Reliable Distrib. Syst. Workshop*, 2015, pp. 13–18.
[17] T. Pitakrat *et al.*, "A comparison of machine learning algorithms for proactive hard disk drive failure detection," in *Proc. 4th Int. ACM Sigsoft Symp. Architecting Critical Syst.*, 2013, pp. 17–21.
[18] J. Zhang *et al.*, "Tier-scrubbing: An adaptive and tiered disk scrubbing scheme with improved MTTD and reduced cost," in *Proc. 57th ACM/EDAC/IEEE Des. Autom. Conf.*, 2020, pp. 1–6.
[19] W. Jiang *et al.*, "Are disks the dominant contributor for storage failures—A comprehensive study of storage subsystem failure characteristics," *ACM Trans. Storage*, vol. 4, no. 3, pp. 7:1–7:25, 2008.
[20] F. L. F. Pereira, F. D. dos Santos Lima, L. G. de Moura Leite, J. P. P. Gomes, and J. de Castro Machado, "Transfer learning for Bayesian networks with application on hard disk drives failure prediction," in *Proc. Brazilian Conf. Intell. Syst.*, 2017, pp. 228–233.
[21] J. Zhang *et al.*, "An end-to-end automatic cloud database tuning system using deep reinforcement learning," in *Proc. Int. Conf. Manage. Data*, 2019, pp. 415–432.
[22] I. V. Tetko *et al.*, "Neural network studies, 1. Comparison of overfitting and overtraining," *J. Chem. Inf. Comput. Sci.*, vol. 35, no. 5, pp. 826–833, 1995.
[23] G. Hamerly and C. Elkan, "Bayesian approaches to failure fredic-tion for disk drives," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, pp. 202–209.
[24] G. F. Hughes, J. F. Murray, K. Kreutz-Delgado, and C. Elkan, "Improved disk-drive failure warnings," *IEEE Trans. Rel.*, vol. 51, no. 3, pp. 350–357, Sep. 2002.
[25] J. F. Murray *et al.*, "Hard drive failure prediction using non-parametric statistical methods," in *Proc. Int. Conf. Artif. Neural Netw.*, 2003, pp. 1–4.
[26] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[27] F. Salfner *et al.*, "A survey of online failure prediction methods," *ACM Comput. Surv.*, vol. 42, no. 3, pp. 10:1–10:42, 2010.
[28] S. E. A. Ganguly, "A practical approach to hard disk failure prediction in cloud platforms: Big data model for failure management in datacenters," in *Proc. IEEE 2nd Int. Conf. Big Data Comput. Service Appl.*, 2016, pp. 105–116.
[29] J. Li *et al.*, "Hard drive failure prediction using classification and regression trees," in *Proc. 44th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, 2014, pp. 383–394.
[30] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, 2001.
[31] J. Li, R. J. Stones, G. Wang, Z. Li, X. Liu, and K. Xiao, "Being accurate is not enough: New metrics for disk failure prediction," in *Proc. 35th IEEE Symp. Reliable Distrib. Syst.*, 2016, pp. 71–80.
[32] J. Li *et al.*, "Hard drive failure prediction using decision trees," *Rel. Eng. Sys. Saf.*, vol. 164, pp. 55–65, 2017.
[33] R. Johnson and T. Zhang, "Learning nonlinear functions using regularized greedy forest," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 942–954, Mar. 2014.
[34] M. M. Botezatu *et al.*, "Predicting disk replacement towards reliable data centers," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 39–48.
[35] C. Xu, G. Wang, X. Liu, D. Guo, and T.-Y. Liu, "Health status assessment and failure prediction for hard drives with recurrent neural networks," *IEEE Trans. Comput.*, vol. 65, no. 11, pp. 3502–3508, Nov. 2016.
[36] T. Mikolov, S. Kombrink, L. Burget, J. Černocký, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2011, pp. 5528–5531.
[37] T. Mikolov, J. Kopecky, L. Burget, O. Glembek, and J. Černocký, "Neural network based language models for highly inflective languages," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2009, pp. 4725–4728.
[38] F. Mahdisoltani *et al.*, "Improving storage system reliability with proactive error prediction," in *Proc. USENIX Conf. Usenix Annu. Tech. Conf.*, 2017, pp. 391–402.
[39] J. T. Zhou *et al.*, "Hybrid heterogeneous transfer learning through deep learning," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 2213–2220.
[40] P. Prettenhofer and B. Stein, "Cross-language text classification using structural correspondence learning," in *Proc. 48th Annu. Meeting Assoc. Comput. Linguistics*, 2010, pp. 1118–1127.
[41] B. Kulis, K. Saenko, and T. Darrell, "What you saw is not what you get: Domain adaptation using asymmetric kernel transforms," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 1785–1792.
[42] M. Harel and S. Mannor, "Learning from multiple outlooks," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 401–408.
[43] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 3169–3176.
[44] K. Sarinnapakorn and M. Kubat, "Combining subclassifiers in text categorization: A DST-based solution and a case study," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 12, pp. 1638–1651, Dec. 2007.
[45] R.-E. Fan *et al.*, "LIBLINEAR: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, 2008.
[46] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[47] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *J. Statist. Planning Inference*, vol. 90, no. 2, pp. 227–244, 2000.
[48] W. Dai *et al.*, "Boosting for transfer learning," in *Proc. 24th Int. Conf. Mach. Learn.*, 2009, pp. 193–200.
[49] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Ann. Math. Statist.*, vol. 22, pp. 79–86, 1951.
[50] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proc. 14th Int. Joint Conf. Artif. Intell.*, 1995, pp. 1137–1143.
[51] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," *Remote Sens. Environ.*, vol. 62, no. 1, pp. 77–89, 1997.
[52] D. Powers, "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation," *J. Mach. Learn. Technol.*, vol. 2, pp. 37–63, Jan. 2007.



Ji Zhang is currently working toward the PhD degree in computer science and technology from the Huazhong University of Science and Technology (*HUST*), Wuhan, China and currently a visiting scholar with the Center for Data Science of New York University, New York. His major is computer system structure. He has been an internship at the Intelligent Cloud Storage Joint Research center of *HUST* and *Tencent*. Currently, his main research interests are using artificial intelligence (AI) technologies to optimize the system of data storage or data management. He has published papers in international conferences and journals including SIGMOD, ICPP, DAC, FAST, TPDS, and NEDB, etc.

Ke Zhou (Member, IEEE) received the BE, ME, and PhD degrees in computer science and technology from the Huazhong University of Science and Technology (*HUST*), China, in 1996, 1999, and 2003, respectively. He is currently a professor of the School of Computer Science and Technology and Wuhan National Laboratory for Optoelectronics, *HUST*. His main research interests include computer architecture, cloud storage, parallel I/O, and storage security. He has more than 50 publications in journals and international conferences, including ACM SIGMOD, ICCP, the *IEEE Transactions on Parallel and Distributed Systems*, the Performance Evaluation (PEVA), FAST, USENIX ATC, MSST, ACM MM, INFOCOM, SYSTOR, MASCOTS, ICCD, etc. He is a member of USENIX.

Ping Huang received the PhD degree from the Huazhong University of Science and Technology, China, in 2013. He is currently a research assistant with the Department of Computer and Information Sciences, Temple University, Philadelphia, Pennsylvania. His main research interest includes non-volatile memory, operating system, distributed systems, DRAM, GPU, Key-value systems, etc. He has published papers in various international conferences and journals, including SYSTOR, NAS, MSST, USENIX ATC, Eurosys, IFIP Performance, INFOCOM, SRDS, MASCOTS, ICCD, the *Journal of Systems Architecture (JSA)*, the Performance Evaluation (PEVA), the Sigmetrics, ICPP, the *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, the *ACM Transactions on Storage*, etc.

Xubin He (Senior Member, IEEE) received the BS and MS degrees in computer science from the Huazhong University of Science and Technology, China, in 1995 and 1997, respectively, and the PhD degree in electrical engineering from the University of Rhode Island, Kingston, Rhode Island, in 2002. He is currently a professor with the Department of Computer and Information Sciences, Temple University, Philadelphia, Pennsylvania. His research interests include computer architecture, data storage systems, virtualization, and high availability computing. He received the Ralph E. Powe Junior Faculty Enhancement Award, in 2004 and the Sigma Xi Research Award (TTU Chapter) in 2005 and 2010. He is a member of the IEEE Computer Society and USENIX.

Ming Xie is currently a general manager in Cloud Architecture Platform Department at *Tencent* Corporation. His main research includes cloud computing, massive data storage, and intelligent operation and maintenance.

Bin Cheng is currently a director in Cloud Architecture Platform Department at *Tencent* Corporation. His main research includes cloud computing, massive data storage, machine learning and database system, etc.

Yongguang Ji is currently a group leader in Cloud Architecture Platform Department at *Tencent* Corporation. His main research includes cloud computing, cloud block storage, I/O systems and performance optimization, etc.

Yinhu Wang is currently a senior staff engineer in Cloud Architecture Platform Department at *Tencent* Corporation. His main research includes distributed systems, massive data storage system, disk failure prediction, etc.