Transfer Learning for Anomaly Detection through Localized and Unsupervised Instance Selection

Vincent Vercruyssen, Wannes Meert, Jesse Davis
DTAI group, KU Leuven, Belgium
firstname.lastname@kuleuven.be

Abstract

Anomaly detection attempts to identify instances that deviate from expected behavior. Constructing performant anomaly detectors on real-world problems often requires some labeled data, which can be difficult and costly to obtain. However, often one considers multiple, related anomaly detection tasks. Therefore, it may be possible to *transfer labeled instances* from a related anomaly detection task to the problem at hand. This paper proposes a novel transfer learning algorithm for anomaly detection that selects and transfers relevant labeled instances from a source anomaly detection task to a target one. Then, it classifies target instances using a novel semi-supervised nearest-neighbors technique that considers both unlabeled target and transferred, labeled source instances. The algorithm outperforms a multitude of state-of-the-art transfer learning methods and unsupervised anomaly detection methods on a large benchmark. Furthermore, it outperforms its rivals on a real-world task of detecting anomalous water usage in retail stores.

1 Introduction

Anomaly or outlier detection is a fundamental data analysis task that involves identifying instances in a dataset that differ from what was expected (Chandola, Banerjee, and Kumar 2009). Anomaly detection is important in practice as anomalies often correspond to substantial problems that could have significant costs, such as abnormal web traffic (Robberechts et al. 2018), or credit card fraud (Chan et al. 1999).

Anomaly detection can naturally be posed as an *unsupervised* learning task (Ramaswamy, Rastogi, and Shim 2000; Breunig et al. 2000). Typically, unsupervised approaches exploit the underlying assumption that anomalies occur infrequently, which means they fall in low-density regions of the instance space, or that anomalies are far away from normal instances to identity them. However, real-world data regularly violate this assumption, degrading the unsupervised approaches' performance (e.g., system maintenance can occur infrequently and irregularly, but is not anomalous). Labeled data offers the possibility to correct the mistakes made by unsupervised detectors. Unfortunately, a *fully supervised* approach to anomaly detection is infeasible due to the fact that collecting examples of real-world anomalies is often expensive (e.g., a machine breaking), meaning that is not a viable strategy to permit anomalous behavior for the sake of data generation. This has spurred interest in *semi-supervised* approaches to anomaly detection, usually in conjunction with active learning to efficiently collect the labels.

Real-world anomaly detection tasks often involve monitoring numerous assets, each of which is only slightly different. Such a situation may arise when monitoring machines in a factory, resource usage in a chain of retail stores, or windturbine farms. These use cases entail monitoring a large number of assets as a big retail chain could have 100s if not 1000s of stores. Therefore, even when using a strategy like active learning, it may be impossible to collect labels for each individual asset. Given that these use cases involve multiple similar anomaly detection tasks, it may be possible to employ *transfer learning* to transfer labeled instances from one task to another. This could then alleviate the need to collect labels for each task separately.

Motivated by these types of applications, this paper proposes LocIT, a novel transfer learning algorithm tailored towards anomaly detection. It works in two steps. First, given a partially labeled source dataset and an unlabeled target dataset, LocIT selects a subset of the labeled source instances to transfer to the target dataset. It picks those instances that have similar localized data distributions in both the source and target dataset. Second, it assigns an anomaly score using a semi-supervised nearest-neighbor approach that considers both the transferred, labeled source instances and the unlabeled target instances. Empirically, LocIT outperforms a multitude of existing transfer learning and anomaly detection methods on a new transfer learning benchmark for anomaly detection. Moreover, it outperforms its competitors on a real-world anomaly detection problem of identifying anomalous water usage in multiple retail stores. Finally, we provide an implementation of LocIT.[1]

2 Preliminaries

Transfer Learning for Anomaly Detection. Transfer learning aims to learn a model for one dataset (the target do-

[1] https://github.com/Vincent-Vercruyssen/LocIT

main) given access to data from a related dataset (the source domain) (Van Haaren, Kolobov, and Davis 2015). As this paper concerns anomaly detection, the task is to assign a score to each instance in the target dataset that quantifies how anomalous it is. We use D_S (D_T) to denote the source (target) dataset. We use x_s (x_t) to refer to an instance from the source (target) dataset.

Three common transfer learning assumptions (Kouw and Loog 2018) apply to the anomaly detection task. First, the source and target data are from the same m-dimensional feature space. Second, the source and target marginal distributions differ (*covariate shift* assumption). This happens when distinct behaviors are observed in either domain. Third, the conditional distributions can differ due to changes in context: the same behavior may have different meanings in the two domains (*concept shift* assumption). The last two assumptions complicate the transfer task.

Nearest Neighbors and KNNO. We use several nearest neighbors concepts. The k-distance of an instance x is the distance to its k^{th} nearest neighbor in a dataset D, and is denoted by k-dist(x, D). The set of x's k nearest neighbors in D is denoted by $N_k(x, D)$. The standard weighting function in distance-weighted KNN is $w(x_i; x_j) = \frac{1}{\delta(x_i,x_j)^2}$, where δ is the Euclidean distance between two instances x_i and x_j. Finally, KNNO ranks all instances in a dataset by their k-distance, with higher distances signifying more anomalous instances (Ramaswamy, Rastogi, and Shim 2000).

Algorithm 1: LOCIT(D_S, D_T, ψ, k)

Input: source data D_S, target data D_T, neighborhood sizes ψ and k
Result: Anomaly score for the instances in D_T

Phase (i): Localized instance-based transfer.
$F_{pos}, F_{neg} = \varnothing$
for $x_t \in D_T$ **do**
  x_n is the nearest neighbor of x_t in $D_T \setminus \{x_t\}$
  x_f is the farthest neighbor of x_t in $D_T \setminus \{x_t\}$
  $N1 = N_\psi(x_t, D_T)$ and $N2 = N_\psi(x_n, D_T)$
  $F_{pos} = F_{pos} \cup [d_1(N1, N2), d_2(N1, N2)]$
  $N2 = N_\psi(x_f, D_T)$
  $F_{neg} = F_{neg} \cup [d_1(N1, N2), d_2(N1, N2)]$
svm $=$ *fitSVM*(F_{pos}, F_{neg})
$D_{trans} = \varnothing$
for $x_s \in D_S$ **do**
  $N1 = N_\psi(x_s, D_S)$ and $N2 = N_\psi(x_s, D_T)$
  $f_s = [d_1(N1, N2), d_2(N1, N2)]$
  if *svm.predict*$(f_s) = pos$ **then**
    $D_{trans} = D_{trans} \cup \{x_s\}$
Phase (ii): Prediction in the target domain.
$D^* = D_T \cup D_{trans}$
for $x_t \in D_T$ **do**
  Predict x_t's anomaly score using Eq. 4, D^* and k

3 The LOCIT Algorithm

The problem we are trying to solve can be defined as:

Given: A (partially) labeled source dataset D_S and an unlabeled target dataset D_T from the same feature space;

Do: Assign an anomaly score to each instance in D_T using D_T and a subset of D_S.

Our novel *localized instance-transfer algorithm* (LOCIT) takes a two-step approach for addressing this task. First, LOCIT decides in a label-agnostic way whether to transfer each source instance to the target domain by checking if the instance's local data distribution is similar in both the source and target domains. LOCIT takes an unsupervised transfer approach because the target labels are not available and the value of the source label should not influence the transfer decision. Second, LOCIT assigns an anomaly score to each target instance by employing a novel *semi-supervised anomaly detection algorithm*. Algorithm 1 details the overall control flow of LOCIT and the following subsections describe each step in more detail.

3.1 Localized Instance-Based Transfer

An instance-transfer function $f(x_s; D_S, D_T) \mapsto \{0, 1\}$ decides whether to transfer each source instance $x_s \in D_S$ to the target domain (1) or not (0). Ideally, each transferred source instance has the same meaning (i.e., would be labeled similarly) in both domains. It only makes sense to transfer a source anomaly (normal) to the target domain if it is similar to a target anomaly (normal). However, LOCIT must make this assessment without access to any labels for the target instances. Consequently, LOCIT makes the intuitive *assumption* that an instance has a similar meaning in both the source and target domain if the *local structure* of the source and target marginal distributions around the instance are similar, where the structure of the distributions is characterized by first and second order statistics.

Characterizing an Instance's Local Structure. For a given source instance x_s, LOCIT defines the *localized source distribution* using the set $N_\psi(x_s, D_S)$ of x_s's ψ nearest neighbors in the source data. Similarly, the *localized target distribution* is based on the set $N_\psi(x_s, D_T)$ of x_s's ψ nearest neighbors in the target data. The assumption is that if the distribution over $N_\psi(x_s, D_S)$ is sufficiently similar to the distribution over $N_\psi(x_s, D_T)$, x_s can be transferred, where the similarity is measured by comparing the following first and second order statistics of $N_\psi(x_s, D_S)$ and $N_\psi(x_s, D_T)$:

Location distance: This is the $l2$-norm of the difference between the centroids (i.e., arithmetic mean) of two neighborhood sets $N1$ and $N2$:

$$d_1(N1, N2) = \left\| \frac{1}{k} \left(\sum_{x_i \in N1} x_i - \sum_{x_j \in N2} x_j \right) \right\|_2 . \quad (1)$$

Here, LOCIT computes $d_1(N_\psi(x_s, D_S), N_\psi(x_s, D_T))$. Intuitively, large values of d_1 indicate less overlap between the regions covered by the two sets, which correspondingly decreases the chance of meaningful transfer.

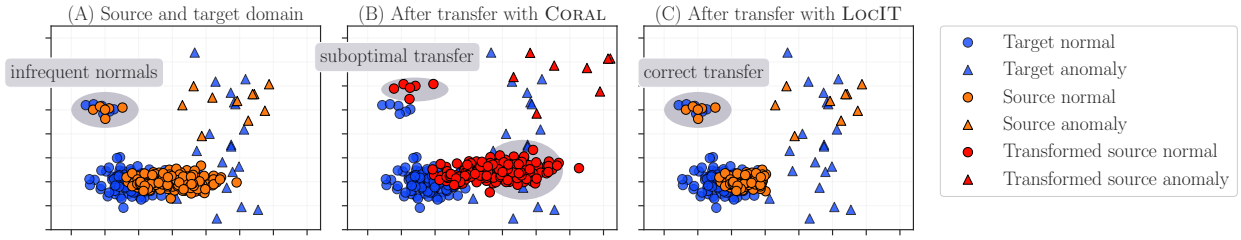

Figure 1: Both the source and target domain (panel A) contain a small (infrequent) cluster of normal instances, which many unsupervised algorithms would incorrectly flag as anomalous. Hence, transferring labeled instances from the source to the target could help. CORAL (panel B) transforms the source data to align the global statistics between the domains. However, outliers skew the correction, resulting in a suboptimal mapping. Here, source normals are mapped to target anomalies and the clusters of infrequent normal instances do not match. LOCIT (panel C), on the other hand, only transfers the subset of source instances for which the *localized* source and target distributions match. Hence, it correctly transfers labeled source instances to the small cluster of infrequent normals, while avoiding incorrect transfer on the lower right.

Correlation distance: This is the relative distance between the covariance matrices of two neighborhood sets:

$$d_2(N1, N2) = \frac{\|C_{N1} - C_{N2}\|_F}{\|C_{N1}\|_F} \quad (2)$$

where $\|\cdot\|_F$ is the Frobenius norm and C is the covariance matrix. Again, LOCIT computes $d_2(N_\psi(x_s, D_S), N_\psi(x_s, D_T))$. If d_2 is large, this signals that the underlying localized source and target distributions are distinct in shape and/or orientation, which again reduces the chance of meaningful transfer.

The size of the neighborhood set ψ is the only hyperparameter in the transfer step of LOCIT. Intuitively, ψ controls the strictness of the instance transfer. If ψ is maximal (the minimum of the number of source or target instances) LOCIT ignores local distributional differences and considers the full global structure of the source and target domains to determine transfer. If ψ is 1, LOCIT only transfers a source point when the distance to its nearest neighbor in the source and target domain is similar to the average distance between any two neighboring points in the target domain.

Learning the Instance-Transfer Function. The instance transfer function f needs to decide whether to transfer x_s by combining the information provided by d_1 and d_2. LOCIT learns an SVM classifier on the target distribution to serve as f. The classifier predicts if a source instance x_s belongs to the target domain by looking at the correlation and location distance between the neighborhood sets of the instance in the source and target data.

To train the classifier, LOCIT generates training data by only considing the target data. It does so by leveraging the smoothness assumption that neighboring target instances should have similar localized distributions while far-away instances should not. Thus, one positive training example is generated for each instance $x_t \in D_T$ by finding its *nearest neighbor* $x_n \in D_T \setminus \{x_t\}$ and computing $d_1(N_\psi(x_t, D_T), N_\psi(x_n, D_T))$ and $d_2(N_\psi(x_t, D_T), N_\psi(x_n, D_T))$. Similarly, the negative training examples are generated by computing for each instance $x_t \in D_T$ a feature vector consisting of the distances between the neighborhood sets of x_t and its *farthest neighbor* $x_f \in D_T \setminus \{x_t\}$.

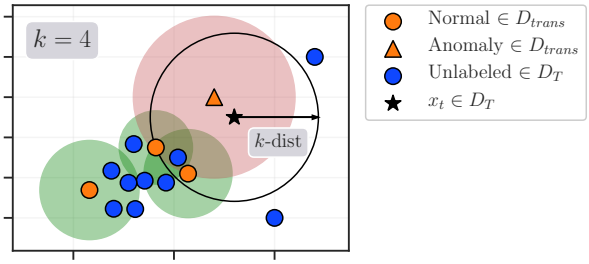

Figure 2: After transfer, SSKNNO computes an anomaly score for each target instance as a weighted combination of an *unsupervised* score and a *supervised* score. In this example using $k = 4$, x_t's nearest neighbors are two source normals, a source anomaly, and an unlabeled target instance. Because x_t does not belong to the neighborhood sets of the two source normals (green circles) in its neigbhorhood set, they are excluded when computing the weight for the supervised component of the score. Thus, the weights of the unsupervised and supervised components are respectively $\frac{3}{4}$ and $\frac{1}{4}$.

LOCIT tunes the SVMs hyperparameters using three-fold cross-validation on the generated training data. It selects either a linear or Gaussian kernel and sets $C \in [0.01, 0.1, 0.5, 1, 10, 100]$ for both kernels and $\sigma \in [0.01, 0.1, 0.5, 1, 10, 100]$ for the Gaussian kernel.

Figure 1 compares LOCIT with a popular global domain alignment strategy, CORAL (Sun, Feng, and Saenko 2016), on a small source and target dataset.

3.2 Prediction in the Target Domain

After transfer, there are two challenges with making predictions for the target instances. First, the target domain now contains a mix of labeled and unlabeled instances. Second, because this is an anomaly detection problem, the known labels in a target instance's neighborhood are not necessarily

informative of its label.

The second contribution of LOCIT is a *semi-supervised anomaly detection method*, SSKNNO, that combines the unlabeled and (transferred) labeled instances to compute an anomaly score for each target instance. On the one hand, it considers the local distribution of the unlabeled target instances when computing the score. On the other hand, it weighs this score by comparing the neighborhoods of the (transferred) labeled instances and the unlabeled instances in the target data.

Let $D^* = D_T \cup D_{trans}$ be the set of instances that includes both the target instances and the transferred source instances. For a given target instance x_t, we find the set of all neighbors $N_k(x_t, D^*)$ and denote its subset of labeled neighbors as $L_k(x_t, D^*)$. The weight W_l of the labeled instances in the neighborhood of x_t is now:

$$W_l(x_t) = \frac{|x_i : x_i \in L_k(x_t, D^*) \wedge x_t \in N_k(x_i, D^*)|}{k}. \quad (3)$$

Intuitively, when assigning an anomaly score to x_t, we only want to consider the label of a transferred source instance if the source instance is similar to x_t. For example, if x_t is an isolated instance, its k-nearest neighbors will be far-away. Even if labeled, such instances will not be very predictive of x_t's label. This is reflected in $W_l(x_t)$, which only considers instances in x_t's neighborhood that also include x_t in their neighborhood.

This weight can now be used to compute the anomaly score $a(x_t)$ for instance x_t as a weighted combination between an unsupervised component, a_u, that considers the local data distribution and a supervised component, a_l, that considers nearby labeled instances:

$$a(x_t) = (1 - W_l(x_t))\, a_u(x_t) + W_l(x_t)\, a_l(x_t). \quad (4)$$

The supervised component of the score a_l is the distance-weighted average of the labels of the instances in the neighborhood of x_t:

$$a_l(x_t) = \frac{\sum_{x_i \in L_k} \mathbb{1}_{x_i = anomaly}(x_i)\; w(x_i; x_t)}{\sum_{x_i \in L_k} w(x_i; x_t)} \quad (5)$$

where w is defined as in Section 2 and $\mathbb{1}_{y_i=1}(x_i)$ is 1 if the expert labeled instance x_i as anomalous. The unsupervised component of the score a_u uses k-dist based on the KNNO algorithm but bounds the distance to $[0, 1]$ using a squashing function from the exponential family:

$$a_u(x_t) = 1 - \exp\left(-\frac{k\text{-dist}(x_t, D^*)^2}{2\gamma^2}\right) \quad (6)$$

where γ is the assumed percentage of anomalies and is set to be the proportion of known anomalies in the source domain. This exponential quashing is a monotone function and higher values still represent more anomalous instances. See Figure 2 for a graphical explanation of SSKNNO.

In extreme cases the weight $W_l(x_t)$ can be zero or one. If none of x_t's neighbors are labeled, L_k is empty and $W_l(x_t)$ becomes zero, reducing Eq. 4 to the scaled KNNO score. This can happen if LOCIT's transfer function selects no instances to be transferred. Conversely, if all of x_t's neighbors are labeled and x_t belongs to the neighborhood set of each of its neighbors, the final anomaly score is then the standard, weighted KNN classifier.

4 Related Work

We discuss the most closely related work in transfer learning, domain adaptation and anomaly detection.

Instance-Based Transfer. Many different types of transfer learning exist (Weiss, Khoshgoftaar, and Wang 2016; Pan and Yang 2010). We focus on instance transfer where weighted source domain instances are used to construct a decision function in the target domain (Mignone et al. 2019). Chattopadhyay et al. (2012) proposed 2SW-MDA and CP-MDA. However, unlike our problem setting CP-MDA requires labeled target data to work, while 2SW-MDA does not work when only instances of one class are labeled in the source domain.

Domain Adaptation. Domain adaptation techniques transform both the source and target data into a new, latent feature space that minimizes the distributional differences while preserving the intrinsic structure in the data. Then, they apply standard classifiers in the newly-found space. To find the latent space, one class of methods, such as TCA (Pan et al. 2011), TJM (Long et al. 2014), and GFK (Gong et al. 2012), corrects only the differences in marginal distributions. A second class of methods, such as JDA (Long et al. 2013) and JGSA (Zhang, Li, and Ogunbona 2017), attempts to correct both the marginal and conditional distributions by computing pseudo-labels for the unlabeled source and target data. Finally, methods such as CORAL (Sun, Feng, and Saenko 2016) align the source and target domains in the original feature space. All these methods use the 1-nearest neighbor classifier for the target data, with the transformed source data as the training set.

LOCIT differs from these approaches in three key ways. First, it implicitly corrects for conditional distribution differences between the source and target domains by observing the densities of the data distributions. Second, it uses a semi-supervised nearest-neighbors style classifier in the target domain, which we will show empirically leads to better performance. Third, LOCIT's target domain nearest-neighbors classifier works even if instances from only one class are transferred because it interpolates between a supervised and unsupervised score. In contrast, the other approaches require that labeled instances from all classes are transferred.

Transfer Learning for Anomaly Detection. Only a handful of papers explore the use of transfer learning for anomaly detection. CBIT (Vercruyssen, Meert, and Davis 2017) selects labeled source instances to transfer using a density-based approach and a cluster-based approach and constructs a 1NN classifier in the target domain based on these instances. Unlike LOCIT, CBIT fails if the source domain contains labels from only one class or if all transferred source instances come from the same class. Andrews et al. (2016) tries to reuse learned image representations across different image datasets. Finally, Xiao et al. (2015) designed a robust one-class transfer learning method. Unlike LOCIT, the latter two approaches require labeled target instances to construct the classifier. The lack of labels in anomaly detection task that motivated our approach prohibits using these approaches.

Unsupervised Anomaly Detection. Unsupervised anomaly detection assumes that anomalies are infrequent and differ-

ent than the normal instances. The three most popular and successful classes of methods in this area are local density-based methods (Papadimitriou et al. 2003; Breunig et al. 2000), k-nearest neighbor detectors (Ramaswamy, Rastogi, and Shim 2000), and isolation methods (Liu, Ting, and Zhou 2008). Extensive empirical evaluations have found that both KNNO (Campos et al. 2016; Goldstein and Uchida 2016) and IFOREST perform very well compared to a number of competitors (Domingues et al. 2018). Interestingly, the aforementioned studies do not directly compare IFOREST with KNNO. In contrast, we propose a new, semi-supervised anomaly detection technique that forms a weighted combination of standard KNN and KNNO. The technique also differs from IFOREST, which cannot handle labeled instances.

5 Benchmark Experimental Evaluation

We address the following four empirical questions:

Q1: How does LOCIT perform compared to state-of-the-art transfer learning and anomaly detection algorithms?

Q2: How does the percentage of labeled source instances affect the transfer learning algorithms' performance?

Q3: How does our nearest-neighbor's approach for classifying target instances affect performance?

Q4: How do the values of hyperparameters ψ and k affect the performance of LOCIT?

The code, elaborated explanations, full parameter settings, and further experiments are available in an online appendix.[2]

5.1 Benchmark and Experimental Setup

Compared Approaches. We compare 12 approaches, which can be divided into three categories:

Baseline anomaly detection algorithms. We consider four standard unsupervised anomaly detection techniques that only consider the target domain data: KNNO (a distance-based outlier detection technique (Ramaswamy, Rastogi, and Shim 2000)), LOF (a density-based outlier detection technique (Breunig et al. 2000)), IFOREST (an ensemble-based outlier detection technique (Liu, Ting, and Zhou 2008)), and HBOS (a histogram-based outlier detection technique (Goldstein and Dengel 2012)).

Baseline transfer learning approaches. We consider eight transfer learning approaches: TRANSFERALL (a naive baseline that transfers all source instances as is to the target domain), CORAL (Sun, Feng, and Saenko 2016), TCA (Pan et al. 2011), GFK (Gong et al. 2012), JDA (Long et al. 2013), TJM (Long et al. 2014), JGSA (Zhang, Li, and Ogunbona 2017), and CBIT (Vercruyssen, Meert, and Davis 2017). After transfer, these methods use a KNN classifier to classify the target instances.

LOCIT. This is our proposed transfer learning approach.

Benchmark Construction. We construct our own benchmark because the current transfer learning benchmarks (e.g., (Long et al. 2013; Sun, Feng, and Saenko 2016; Pan et al. 2011)) do not contain anomalies, while the current anomaly detection benchmarks (Campos et al. 2016; Goldstein and Uchida 2016) do not contain source-target pairs for each problem. Our benchmark should display three characteristics. First, for empirical evaluation purposes, the benchmark should contain a large number of source-target domain pairs that have varying degrees of differences between their joint data distributions. Second, each source-target domain pair should live in the same feature space. We start from one of 12 publicly available, multi-class *master datasets*. To generate a target domain, we sample normal (anomalous) target instances from the largest (second largest) class in the master dataset. To ensure distributional differences between the source and the target, we construct source domains by sampling source anomalies and normals from either the same classes or different classes than used in the target domain. We generate 56 unique source-target pairs in this manner. Third, the source and target domains should contain normal and anomalous instances that are nontrivial to classify. We follow Emmott et al.'s procedure (2013) to satisfy this condition.

Experimental Setup. We employ the standard transfer learning experimental setup used in previous work (Long et al. 2013; Zhang, Li, and Ogunbona 2017). For each of the 56 source-target pairs in the benchmark, the following two steps are performed. First, each method transforms the source data and/or selects the source instances to transfer to the target domain. Second, a final classifier is learned using the transferred, labeled source data (and the unlabeled target data) and a prediction is made for each target instance. The anomaly detection algorithms are simply run on the target data (i.e., they do not use any source data). The final classifier's performance is evaluated using the area under the ROC curve (AUROC), as is standard in anomaly detection (Emmott et al. 2013).

Hyperparameter tuning using cross-validation is impossible because there are no labels in the target domain and the distribution of the source data is different (Pan et al. 2011). We simply use the baselines with the hyperparameters recommended in the original papers or in comparative studies. LOCIT has two hyperparameters. We set the neighborhood size $\psi = 20$ and $k = 10$ in SSKNNO. **Q4** analyzes the impact of ψ and k on LOCIT's performance. Further details on the hyperparameters can be found in the online Appendix.

5.2 Experimental Results and Discussion

Q1: LOCIT vs. State-of-the-art. Table 1 compares LOCIT to all baselines when the source domain is fully labeled. LOCIT outperforms all baselines on the full benchmark, having the lowest average AUROC rank (lower is better) and achieving the highest average AUROC.

Furthermore, LOCIT yields an average increase in AUROC of at least $>9\%$ compared to performing unsupervised anomaly detection. This provides evidence that LOCIT's approach to transfer can help in anomaly detection tasks. Finally, LOCIT achieves average AUROC gains of $>18\%$ over all the transfer learning baselines, indicating it that substantially outperforms the state-of-the-art transfer approaches.

[2] https://github.com/Vincent-Vercruyssen/LocIT

Table 1: Comparison of LOCIT to the state-of-the-art baselines when the source domains are fully labeled. The table shows over the full benchmark: the average AUROC rank ± standard deviation (SD) of each method; the average AUROC ± SD of each method; the number of times LOCIT wins (higher AUROC), draws (absolute difference in AUROC < 0.005), and loses (lower AUROC) vs. each method; and the average percentage change in AUROC ± SD of using LOCIT over each method.

Transfer method	Final classifier	Average AUROC rank ± SD of each method	Average AUROC ± SD of each method	# times LOCIT			Average % change in AUROC using LOCIT over all datasets
				wins	draws	loses	
LOCIT	SSKNNO	**3.741 ± 3.318**	**0.762 ± 0.182**	-	-	-	-
-	KNNO	5.321 ± 3.663	0.705 ± 0.177	**44**	4	8	+9.63% ± 18.31%
CORAL	KNN	5.848 ± 3.311	0.666 ± 0.188	**36**	1	19	+20.65% ± 40.60%
-	LOF	6.018 ± 4.413	0.677 ± 0.113	**37**	0	19	+14.53% ± 31.53%
TRANSFERALL	KNN	6.732 ± 3.646	0.649 ± 0.185	**37**	0	19	+27.34% ± 62.98%
-	IFOREST	6.795 ± 3.898	0.690 ± 0.169	**50**	1	5	+11.32% ± 13.53%
GFK	KNN	7.125 ± 3.312	0.642 ± 0.186	**38**	1	17	+29.10% ± 67.03%
TCA	KNN	7.348 ± 2.356	0.656 ± 0.154	**44**	1	11	+18.20% ± 27.64%
-	HBOS	7.920 ± 4.015	0.646 ± 0.216	**48**	2	6	+29.16% ± 53.65%
TJM	KNN	8.000 ± 2.793	0.627 ± 0.175	**44**	2	10	+27.22% ± 39.12%
CBIT	KNN	8.223 ± 2.345	0.623 ± 0.150	**46**	0	10	+24.57% ± 28.54%
JDA	KNN	8.509 ± 2.910	0.617 ± 0.170	**44**	0	12	+30.51% ± 48.76%
JGSA	KNN	9.420 ± 3.845	0.576 ± 0.096	**46**	0	10	+33.99% ± 33.27%

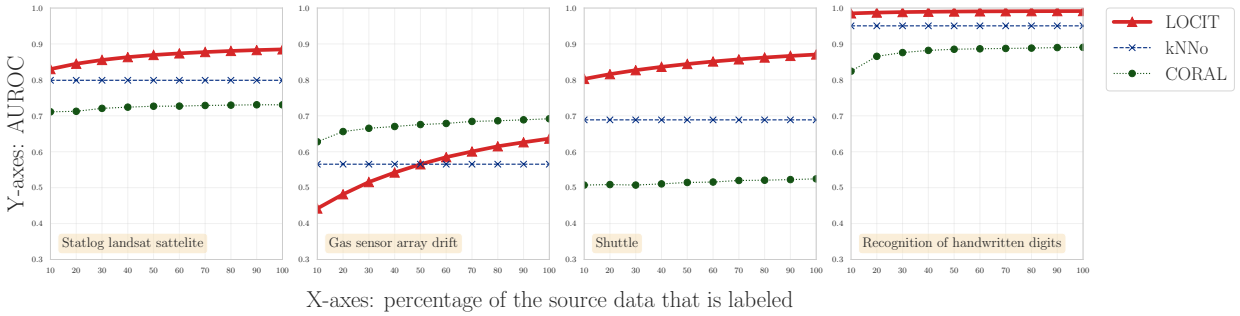

Figure 3: The AUROC averaged over all source-target pairs on four representative master datasets, as a function of the percentage of labeled source data for LOCIT, KNNO, and CORAL. LOCIT's and CORAL's performance improves as labels are added to the source domain. With 50% of the source labels, LOCIT always performs better or equivalently to KNNO.

Q2: Effect of Varying the Percentage of Source Labels. To explore how the amount of labeled source data affects performance in the target domain, we vary the proportion of labeled sources instances. This is relevant because often it is difficult to obtain fully labeled data for anomaly detection problems. We randomly sampled $10\% - 100\%$ with increments of 10% of the source instances and only considered the labels of these instances when performing transfer. We repeated this procedure five times and averaged results.

Figure 3 shows how the average AUROC varies as a function of the proportion of labeled source instances on four representative master datasets. For readability, the plots show the results for LOCIT as well as CORAL and KNNO, which are the best transfer and anomaly detection baselines respectively as determined by Table 1. KNNO's curves are straight lines because it only considers the target data. CORAL improves as more source labels become available. Regardless of the proportion of source labels, LOCIT outperforms CORAL on nine of the 12 master datasets. Furthermore, with only 10% of the source labels, LOCIT performs better than or similar to KNNO on eight of the 12 master datasets. As more source labels become available, LOCIT's performance improves and with 50% of the source labels, LOCIT always performs better or equivalently to KNNO. As source labels continue to be added, the performance gap with KNNO widens.

Table 2: LOCIT vs. the best transfer baselines from Table 1 coupled to use SSKNNO as the target domain classifier. The table shows the average AUROC ± SD and the average percentage change in AUROC ± SD of using LOCIT over each competitor for fully labeled source domains.

Transfer method + SSKNNO	Average AUROC ± SD of each method	Average % change in AUROC using LOCIT over all datasets
LOCIT	**0.762 ± 0.182**	-
CORAL	0.735 ± 0.171	+3.88% ± 9.54%
TRANSFERALL	0.727 ± 0.173	+5.17% ± 10.29%
GFK	0.705 ± 0.172	+8.95% ± 13.28%
CBIT	0.713 ± 0.171	+7.28% ± 10.64%
TCA	0.533 ± 0.177	+74.81% ± 116.18%

Q3: Impact of the SSKNNO Approach. To assess how much of LOCIT's gains come from using our novel SSKNNO approach, we use SSKNNO instead of KNN as the target domain classifier for the best competing transfer methods from Table 1. Table 2 shows the results for this

Table 3: The average AUROC ± SD on the real-world water usage data by LOCIT, the best non-transfer baseline (KNNO) and the two best transfer baselines (CORAL and TRANSFERALL) on the benchmark. Best results are in bold. LOCIT consistently outperforms its competitors.

source	target	KNNO	TRANSFERALL	CORAL	LOCIT
store 1	store 2	0.779 ± 0.126	0.611 ± 0.154	0.646 ± 0.133	$\mathbf{0.800 \pm 0.112}$
store 1	store 3	0.943 ± 0.063	0.758 ± 0.192	0.776 ± 0.182	$\mathbf{0.969 \pm 0.049}$
store 2	store 1	0.754 ± 0.227	0.771 ± 0.207	0.771 ± 0.219	$\mathbf{0.779 \pm 0.227}$
store 2	store 3	0.943 ± 0.063	0.790 ± 0.232	0.794 ± 0.232	$\mathbf{0.969 \pm 0.050}$
store 3	store 1	0.754 ± 0.227	0.704 ± 0.159	0.702 ± 0.159	$\mathbf{0.779 \pm 0.227}$
store 3	store 2	0.779 ± 0.126	0.592 ± 0.164	0.638 ± 0.135	$\mathbf{0.800 \pm 0.112}$

experiment. For CORAL, GFK, TRANSFERALL, and CBIT, using SSKNNO as the classification approach results in improved performance versus the KNN classifier. The exception is TCA. Even when they are coupled with the SSKNNO classifier, LOCIT outperforms all the baselines. It performs better or similar (difference in AUROC < 0.005) than all competitors on at least 37 out of 56 benchmark datasets, and achieves a higher average AUROC than all other methods. This indicates that both LOCIT's novel instance selection procedure and its approach to classification in the target domain contribute to its superior performance.

Q4: Impact of LOCIT's Hyperparameters. LOCIT has two hyperparameters. First, the hyperparameter ψ controls the strictness of the instance-selection step. Choosing $\psi \geq 20$ generally yields good performance. Lower values degrade performance. Second, hyperparameter k in the SSKNNO step of LOCIT controls the number of neighbors considered when deriving an anomaly score for an instance. A *good* value of k depends on the type of dataset. Lower values of $k < 20$ are an overall good choice across a range of datasets. See Figure 6 in the online Appendix.

6 Real-world Experimental Evaluation

We evaluate LOCIT's effectiveness on a real-world transfer task of detecting anomalous water usage in a chain of retail stores. The retail company operates hundreds of stores and wants to avoid excess usage due its harmful environment impact and to minimize costs. Detecting anomalous usage is challenging because the data contains infrequent, but *normal* irregularities such as maintenance patterns, after-hour events, and temporary alterations in opening hours. Similarly, certain *abnormal* behaviors such as leaks occur relatively frequently. Hence, the data violate the standard assumptions made in unsupervised anomaly detectors.

Having access to some labeled examples could help improve detection performance and correct the mistakes made by unsupervised detectors. Unfortunately, it is not feasible to label even a small subset of the data for every store due to the time costs associated with labeling. This raises the following the question:

> Can labeled instances be transferred between different scores to improve anomaly detection performance?

However, transfer in this setting is not straightforward because the observed usage, and consequently what constitutes (ab)normal behavior, varies substantial due to contextual differences among stores (e.g., location, size, clientele, opening hours, services offered, etc.).

6.1 Data and Methodology

We have three full years of historical water usage time series data for three stores. The data consists of a univariate measurement that is recorded every five minutes. In each store, company experts labeled about *ten percent* of the data, each of the provided labels indicating whether a block of one hour (e.g., 01:00-02:00) shows normal behavior or not. The rest of the data is unlabeled.

In each store, the time series data is first divided into non-overlapping one-hour windows (i.e., 00:00-01:00, 01:00-02:00, etc.). Then, each window is transformed into a length 31 feature vector describing the signal's characteristics during that window (Vercruyssen et al. 2018). Because of a store's opening hours, the time of day has a significant effect on water usage. Therefore, all windows for the same hour interval (e.g., 11:00-12:00) are grouped, resulting in 24 groups per store. A separate anomaly detector is trained for each group.

In the transfer learning experiment, each window group for one store is treated as the *unlabeled* target domain, while each of the 24 window groups from a different store serves as the *partially labeled* source domain (about ten percent of the instances is labeled). Having three stores, we construct six unique source-target store combinations, each of which has 576 source-target pairs.

6.2 Experimental Results and Discussion

Table 3 reports the AUROCs of this experiment, averaged per source-target store combination. In aggregate, LOCIT is always better than both KNNO and CORAL on all six store-store combinations. On all 3456 transfer tasks (576 tasks per store-store pair $\times$ six pairs), LOCIT wins (difference in AUROC > 0.01) 2427 and ties (difference in AUROC < 0.01) 280 times with CORAL. It wins 2512 and ties 272 times with TRANSFERALL. Finally, it wins 1868 and ties 1441 times with KNNO. LOCIT outperforms or ties with KNNO on 95% of the real-world transfer tasks, which provides evidence of the effectiveness of label transfer to improve anomaly detection in a real-world setting.

7 Conclusions

While anomaly detection would benefit from labeled data, it is often done in an unsupervised manner because acquiring labels in practice, particularly for anomalies, can be difficult and costly. We considered using transfer learning to acquire labeled instances from a different, but related anomaly detection task. We proposed a novel instance-based transfer method for anomaly detection. Empirically, we have shown that it outperforms numerous unsupervised and transfer learning approaches on a large benchmark. Morever, we showcased its ability to outperform its competitors on real-world anomaly detection task of monitoring water usage in multiple retail stores of a large retail company.

Acknowledgements. This work is supported by the Research Foundation Flanders under EOS No. 30992574 (JD, WM); the KU Leuven Research Fund (C14/17/070) (JD); the VLAIO-SBO grant HYMOP (150033) (JD, WM, VV); and received funding from the Flemish Government under the *"Onderzoeksprogramma Artificile Intelligentie (AI) Vlaanderen"* programme (JD, WM, VV). The authors thank the Colruyt Group for providing the use case data.

References

Andrews, J. T.; Tanay, T.; Morton, E. J.; and Griffin, L. D. 2016. Transfer representation-learning for anomaly detection. In *33rd International Conference on Machine Learning*. JMLR.

Breunig, M. M.; Kriegel, H.-P.; Ng, R. T.; and Sander, J. 2000. LOF: Identifying density-based local outliers. *ACM SIGMOD Record* 29(2):93–104.

Campos, G. O.; Zimek, A.; Sander, J.; Campello, R. J.; Micenková, B.; Schubert, E.; Assent, I.; and Houle, M. E. 2016. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery* 30(4):891–927.

Chan, P. K.; Fan, W.; Prodromidis, A. L.; and Stolfo, S. J. 1999. Distributed data mining in credit card fraud detection. *IEEE Intelligent Systems* 14(6):67–74.

Chandola, V.; Banerjee, A.; and Kumar, V. 2009. Anomaly detection: A survey. *ACM computing surveys* 41(3):1–72.

Chattopadhyay, R.; Sun, Q.; Fan, W.; Davidson, I.; Panchanathan, S.; and Ye, J. 2012. Multisource domain adaptation and its application to early detection of fatigue. *ACM Transactions on Knowledge Discovery from Data* 6(4):18.

Domingues, R.; Filippone, M.; Michiardi, P.; and Zouaoui, J. 2018. A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern Recognition* 74:406–421.

Emmott, A. F.; Das, S.; Dietterich, T.; Fern, A.; and Wong, W.-K. 2013. Systematic construction of anomaly detection benchmarks from real data. In *ACM SIGKDD Workshop on Outlier Detection and Description*, 16–21.

Goldstein, M., and Dengel, A. 2012. Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm. *KI-2012: Poster and Demo Track* 59–63.

Goldstein, M., and Uchida, S. 2016. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one* 11(4):e0152173.

Gong, B.; Shi, Y.; Sha, F.; and Grauman, K. 2012. Geodesic flow kernel for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, 2066–2073. IEEE.

Kouw, W. M., and Loog, M. 2018. An introduction to domain adaptation and transfer learning. *arXiv:1812.11806*.

Liu, F. T.; Ting, K. M.; and Zhou, Z.-H. 2008. Isolation forest. In *8th International Conference on Data Mining*, 413–422. IEEE.

Long, M.; Wang, J.; Ding, G.; Sun, J.; and Yu, P. S. 2013. Transfer feature learning with joint distribution adaptation. In *Conference on Computer Vision and Pattern Recognition*, 2200–2207. IEEE.

Long, M.; Wang, J.; Ding, G.; Sun, J.; and Yu, P. S. 2014. Transfer joint matching for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, 1410–1417. IEEE.

Mignone, P.; Pio, G.; DElia, D.; and Ceci, M. 2019. Exploiting transfer learning for the reconstruction of the human gene regulatory network. *Bioinformatics*.

Pan, S. J., and Yang, Q. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22(10):1345–1359.

Pan, S. J.; Tsang, I. W.; Kwok, J. T.; and Yang, Q. 2011. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks* 22(2):199–210.

Papadimitriou, S.; Kitagawa, H.; Gibbons, P. B.; and Faloutsos, C. 2003. Loci: Fast outlier detection using the local correlation integral. In *19th International Conference on Data Engineering*, 315–326. IEEE.

Ramaswamy, S.; Rastogi, R.; and Shim, K. 2000. Efficient algorithms for mining outliers from large data sets. *ACM SIGMOD Record* 29(2):427–438.

Robberechts, P.; Bosteels, M.; Davis, J.; and Meert, W. 2018. Query log analysis: Detecting anomalies in dns traffic at a tld resolver. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 55–67. Springer.

Sun, B.; Feng, J.; and Saenko, K. 2016. Return of frustratingly easy domain adaptation. In *30th AAAI Conference on Artificial Intelligence*, volume 6, 2058–2065.

Van Haaren, J.; Kolobov, A.; and Davis, J. 2015. TODTLER: Two-order-deep transfer learning. In *29th AAAI Conference on Artificial Intelligence*, 3007–3015.

Vercruyssen, V.; Wannes, M.; Gust, V.; Koen, M.; Ruben, B.; and Jesse, D. 2018. Semi-supervised anomaly detection with an application to water analytics. In *International Conference on Data Mining*. IEEE.

Vercruyssen, V.; Meert, W.; and Davis, J. 2017. Transfer learning for time series anomaly detection. In *CEUR Workshop Proceedings*, volume 1924, 27–37.

Weiss, K.; Khoshgoftaar, T.; and Wang, D. 2016. A survey of transfer learning. *Journal of Big Data* 3(9):1–40.

Xiao, Y.; Liu, B.; Philip, S. Y.; and Hao, Z. 2015. A robust one-class transfer learning method with uncertain data. *Knowledge and Information Systems* 44(2):407–438.

Zhang, J.; Li, W.; and Ogunbona, P. 2017. Joint geometrical and statistical alignment for visual domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, 1859–1867. IEEE.

Table 4: Characteristics of the 12 master datasets used to construct the transfer learning for anomaly detection benchmark. The datasets are obtained from the UCI machine learning repository[3]. For each dataset, only the numeric features are used. Age$^{(1)}$ is divided into three classes. The three$^{(2)}$ smallest classes are removed.

Dataset	# Features	# Instances	# Classes
Abalone	8	4177	$3^{(1)}$
Covertypes	54	581012	7
Gas sensor array drift	128	13910	6
Gesture phase segmentation	32	9900	5
Recognition of handwritten digits	64	5620	9
Statlog landsat satellite	36	6435	6
Letter recognition	16	20000	26
Pen-based recognition of digits	16	10992	10
Image segmentation	19	2310	7
Shuttle	9	58000	7
Waveform	21	5000	3
Poker	11	1025010	$7^{(2)}$

8 Appendix

8.1 Benchmark Construction

An appropriate benchmark for transfer learning for anomaly detection should exhibit three characteristics. First, it should contain a sufficient number of datasets, each dataset consisting of a source-target domain pair. The source and target domains contain instances that are sampled from the same m-dimensional instance space (see the first transfer learning assumption in Section 2). Second, each source and target domain should contain normal and anomalous instances, a number of which are nontrivial to classify. The importance of this condition in general anomaly detection benchmarks is stated in (Emmott et al. 2013). Third, the benchmark datasets should portray varying degrees of differences between the joint data distributions of the source and target domain (see the second and third transfer learning assumptions in Section 2).

Our benchmark is constructed to reflect these three characteristics. We start with 12 publicly available, multi-class *master datasets* that are listed in Table 4. Each master dataset is converted into three (or five) unique source-target domain pairs, depending on wether the master dataset contains three (or more) classes. Because each source-target domain pair is constructed from the same master dataset, the source and target instances live in the same m-dimensional instance space. This results in a total of 56 unique benchmark datasets.

For any source-target domain pair, the target domain is constructed by sampling the normal (anomalous) instances from the largest (second largest) class in the master dataset. We ensure that the target domain contains instances that are nontrivial to classify by varying the *point difficulty* of the sampled instances, as outlined in (Emmott et al. 2013). The point difficulty simply quantifies how difficult it is to separate the instances of the two classes in a binary classification problem with full access to the label information. Meaningful source normals and anomalies are sampled from the chosen classes using the same procedure as in the target domain.

Table 5: The average AUROC for each master dataset as a function of the difficulty of the transfer task. The table shows results for LOCIT as well as KNNO and CORAL, which are the best anomaly detection and transfer learning baselines from Table 1. The table is complementary to the transfer difficulty plots in Figure 5. The results show that when the source and target domains are more similar (lower transfer difficulty), transfer yields larger gains. As the source and target domains become more dissimilar, the usefulness of transfer decreases. This decrease, however, is less pronounced for LOCIT than for CORAL.

Dataset	Method	Transfer difficulty level				
		1	**2**	**3**	**4**	**5**
Abalone	LOCIT	0.752	-	**0.584**	**0.404**	-
	CORAL	**0.853**	-	0.578	0.316	-
	KNNO	0.579	-	0.579	0.579	-
Covertypes	LOCIT	0.695	**0.688**	0.704	**0.667**	**0.667**
	CORAL	**0.735**	0.639	**0.724**	0.583	0.554
	KNNO	0.636	0.636	0.636	0.636	0.636
Gas sensor array drift	LOCIT	0.837	0.621	0.72	**0.622**	0.382
	CORAL	**0.974**	**0.674**	**0.887**	0.495	0.429
	KNNO	0.434	0.434	0.434	0.434	**0.434**
Gesture phase segmentation	LOCIT	0.484	0.405	0.428	0.432	0.382
	CORAL	**0.74**	**0.452**	**0.517**	**0.538**	**0.403**
	KNNO	0.392	0.392	0.392	0.392	0.392
Recognition of handwritten digits	LOCIT	**0.996**	**0.996**	**0.995**	**0.989**	**0.981**
	CORAL	0.975	0.824	0.95	0.953	0.751
	KNNO	0.978	0.978	0.978	0.978	0.978
Statlog landsat sattelite	LOCIT	0.934	**0.873**	**0.903**	**0.901**	**0.813**
	CORAL	**0.957**	0.869	0.669	0.507	0.651
	KNNO	0.792	0.792	0.792	0.792	0.792
Letter recognition	LOCIT	0.926	**0.915**	**0.91**	**0.887**	0.796
	CORAL	**0.947**	0.59	0.845	0.652	0.477
	KNNO	0.825	0.825	0.825	0.825	**0.825**
Pen-based recognition of digits	LOCIT	**0.969**	**0.969**	**0.957**	**0.954**	0.93
	CORAL	0.918	0.689	0.837	0.7	0.609
	KNNO	0.932	0.932	0.932	0.932	**0.932**
Poker	LOCIT	**0.696**	**0.696**	**0.615**	0.506	0.51
	CORAL	0.605	0.611	0.589	0.559	0.554
	KNNO	0.612	0.612	0.612	**0.612**	**0.612**
Image segmentation	LOCIT	0.808	**0.772**	**0.781**	0.698	0.627
	CORAL	**0.876**	0.478	0.431	0.517	0.469
	KNNO	0.703	0.703	0.703	**0.703**	**0.703**
Shuttle	LOCIT	**0.933**	**0.917**	**0.895**	**0.818**	**0.787**
	CORAL	0.891	0.699	0.302	0.427	0.303
	KNNO	0.765	0.765	0.765	0.765	0.765
Waveform	LOCIT	**0.852**	-	0.846	**0.865**	-
	CORAL	0.881	-	**0.893**	0.766	-
	KNNO	0.795	-	0.795	0.795	-

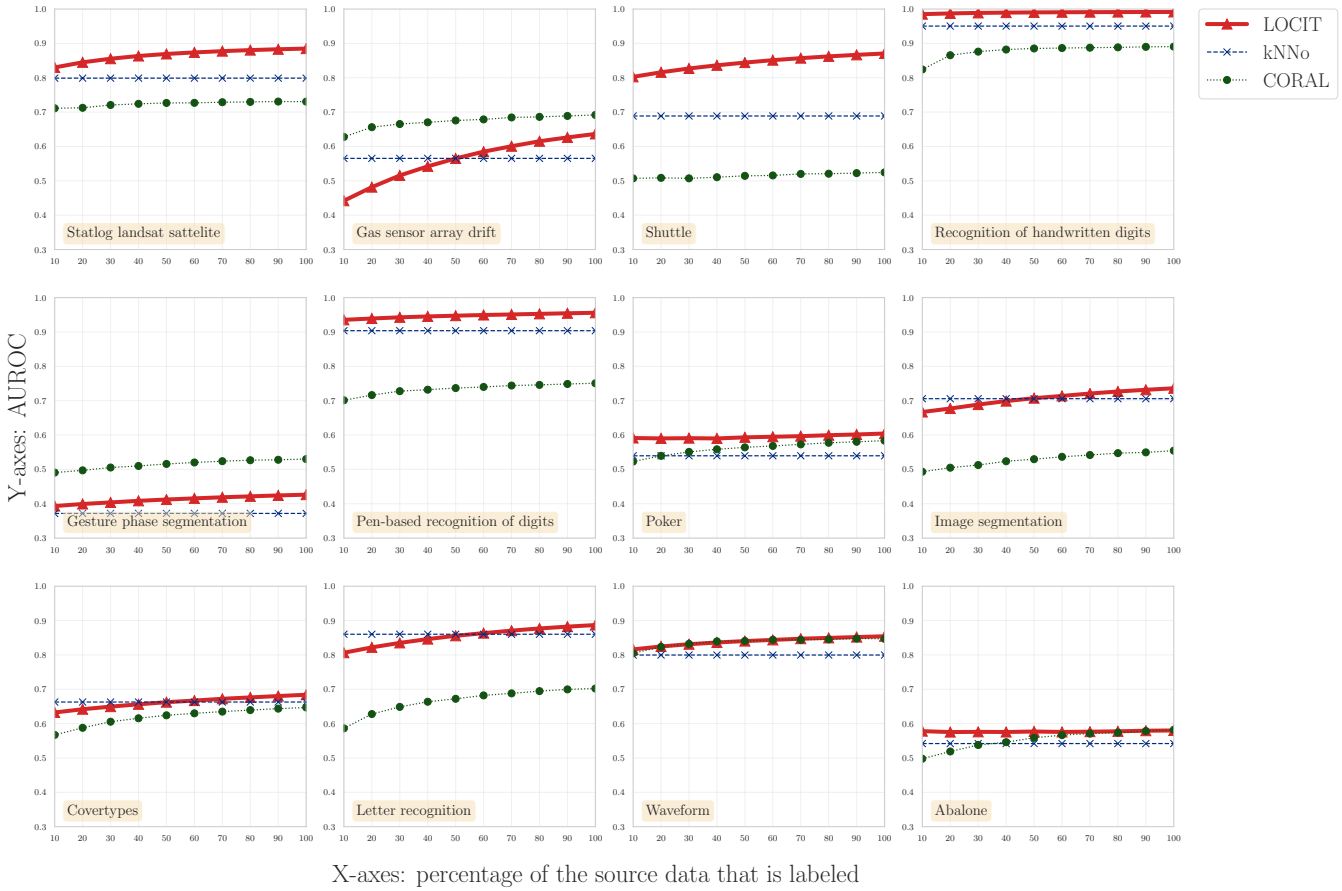

Figure 4: The AUROC averaged over all source-target pairs of each master dataset as a function of the percentage of labels in the source domain. The plot shows results for LOCIT (red line) as well as KNNO (blue line), and CORAL (green line) which are the best anomaly detection and transfer learning baselines from Table 1. LOCIT's and CORAL's performance improves as labels are added to the source domain. When 10% of the source labels are availabe, LOCIT performs better than or similar to KNNO on eight of the 12 master datasets. With 50% of the source labels LOCIT always performs better or equivalently to KNNO. Compared to CORAL, LOCIT always beats it on 10 of the 12 master datasets.

Table 6: Comparison of LOCIT to the best transfer baselines of Table 1 coupled with LOCIT's nearest neighbor approach for making predictions in the target domain. The source domains are fully labeled. The table shows over the full benchmark: the average AUROC rank $\pm$ SD of each method; the average AUROC $\pm$ SD of each method; the number of times LOCIT wins (higher AUROC), draws (absolute difference in AUROC < 0.005), and loses (lower AUROC) vs. each method; and the average percentage change in AUROC $\pm$ SD of using LOCIT over each method.

Transfer method	Final classifier	Average AUROC rank $\pm$ SD of each method	Average AUROC $\pm$ SD of each method	# times LOCIT			Average % change in AUROC using LOCIT over all datasets
				wins	draws	loses	
LOCIT	SSKNNO	$\mathbf{2.071 \pm 1.431}$	$\mathbf{0.762 \pm 0.182}$	-	-	-	-
CORAL	SSKNNO	2.741 ± 0.973	0.735 ± 0.171	**37**	5	14	$+3.88\% \pm 9.54\%$
TRANSFERALL	SSKNNO	3.420 ± 1.047	0.727 ± 0.173	**41**	4	11	$+5.17\% \pm 10.29\%$
GFK	SSKNNO	3.848 ± 1.526	0.705 ± 0.172	**44**	5	7	$+8.95\% \pm 13.28\%$
CBIT	SSKNNO	4.214 ± 1.021	0.713 ± 0.171	**50**	1	5	$+7.28\% \pm 10.64\%$
TCA	SSKNNO	4.705 ± 2.063	0.533 ± 0.177	**42**	0	14	$+74.81\% \pm 116.18\%$

By varying the master dataset classes from which the source normals and anomalies are sampled, we can introduce meaningful distributional differences between the source and target domain. If the source normals (anomalies) are sampled from a different class than the target normals (anomalies), the marginal distributions of both domains will differ. If the source anomalies (normals) and the target normals (anomalies) are drawn from the same class, conditional distribution

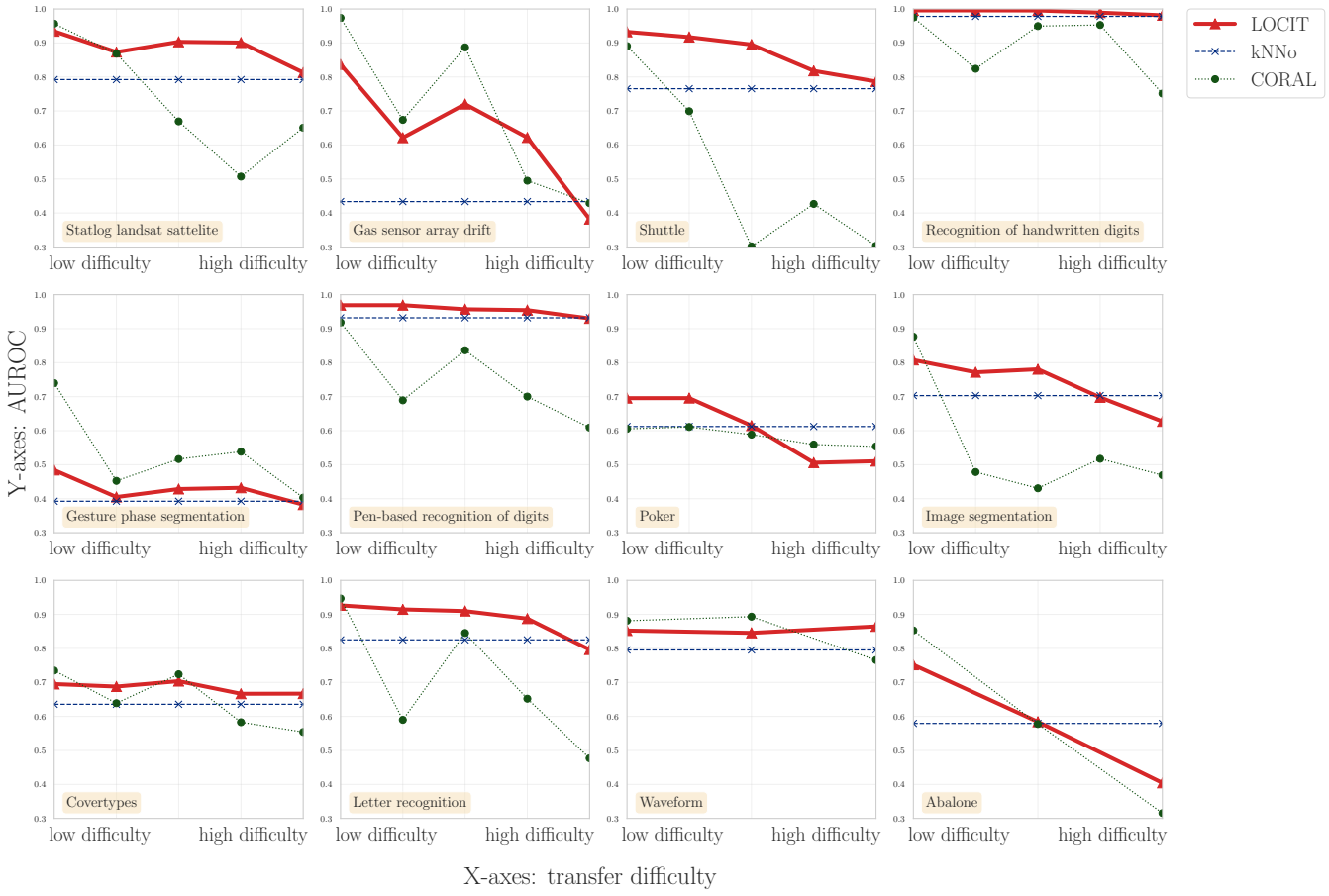

Figure 5: The average AUROC for each master dataset as a function of the difficulty of the transfer task. The plot shows results for LOCIT (red line) as well as KNNO (blue line), and CORAL (green line) which are the best anomaly detection and transfer learning baselines from Table 1. The plots show that when the source and target domains are more similar (lower transfer difficulty), transfer yields larger gains. As the source and target domains become more dissimilar, the usefulness of transfer decreases.

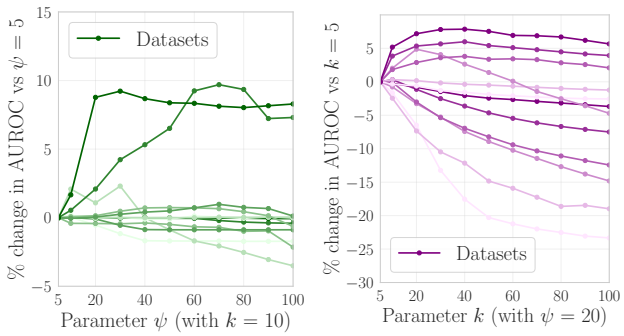

Figure 6: The impact of the hyperparameter ψ in the instance-selection step and the hyperparameter k in the prediction step on LOCIT's performance. For each of the 12 master datasets: (LEFT) the % change in AUROC for increasing values for ψ versus using $\psi = 5$, and (RIGHT) the % change in AUROC for increasing values for k versus using $k = 5$.

differences arise. Both actions decrease the similarity between the source and target domains and make transfer more difficult.

Finally, each source or target domain has 500 normal instances and 50 anomalies. Because the sampling is non-deterministic, we repeat it 10 times for source-target domain pair, and report the averaged results.

8.2 Benchmark Experimental Evaluation

This section contains additional information on the experiments section. First, the hyperparameter settings for the compared approaches are discussed. Then, we go on to discuss the impact of the transfer difficulty on the LOCIT's performance. In addition, the appendix contains the full experimental results for experiment **Q2**. Figure 4 extends Figure 3 to the full benchmark, while Table 7 contains the results of Figure 4 in tabular format.

Baseline Parameter Settings. Because there are no labels in the target domain and the distribution of the source data

is different, hyperparameter tuning using cross-validation is impossible (Long et al. 2013; Gong et al. 2012; Pan et al. 2011). Hence, we resort to setting the hyperparameters of the baselines in accordance with the recommendations made in the original papers or subsequent comparison studies. TRANSFERALL and CORAL have no parameters to set. For TCA, JDA, GFK, TJM, CBIT, and JGSA, we use the parameter settings from the original papers. After transfer, the above methods use the KNN classifier with $k = 10$ to make predictions in the target domain. IFOREST is trained with 100 members in the ensemble. KNNO and LOF both have a single parameter k, the number of neighbors. We set $k = 10$ for KNNO based a recent study that tested KNNO on approximately 1000 datasets with different values for k (Campos et al. 2016). Similarly, we set $k = 10$ for LOF. For HBOS, we set the number of bins to 10.

Impact of Transfer Difficulty on LOCIT. Figure 5 shows the average AUROC values for LOCIT, CORAL (the best transfer method from Table 1) and KNNO (the best anomaly detection method from Table 1) as a function of the transfer difficulty for the 12 master datasets. Table 5 contains the same results but in tabular format. The transfer difficulty is reflective of the similarity between the source and target domains: the more similar the domains, the lower the transfer difficulty. Because KNNO does not use source examples, the difficulty of the transfer task does not affect its performance. However, it is useful to show its performance to assess under what condition transfer is helpful. For each dataset, LOCIT's AUROC is higher than that of kNNO when the difficulty of the transfer task is low. This illustrates an important empirical insight: the gains from transfer increase when the source and target domain are more similar. As the domains become less similar, the performance of the two transfer methods degrades. Still, on nine of the 12 datasets LOCIT performs similarly or beter than KNNO on the most difficult transfer setting. Additionally, on nine of the 12 master datasets, LOCIT consistenly achieves higher AUROCs than CORAL when the transfer difficulty is high. LOCIT's localized instance-selection step helps deal with dissimilarities between the source and target domains, sometimes at the cost of slightly worse performance when domains are similar.

8.3 Real-world Experimental Evaluation

Water Usage Data and Feature Construction. The real-world dataset consists of water usage time series data for three retail stores of a large retail company. The goal of the company is to detect anomalous water usage. An effective detection system will both reduce water consumption and reduce the costs for the company. In each store, water usage was continuously monitored during a full three year period. The time series data consist of a univariate measurement (cubic meters of water consumption) that is recorded every five minutes using a water meter. The data do not contain missing values nor erroneous measurements.

In each store, the time series data are segmented into non-overlapping one-hour windows (i.e., 00:00-01:00, 02:00-03:00, . . . , 23:00-00:00). Then, each window is transformed into a feature vector of length 31, containing the following features:

Summary statistics: mean, maximum, minimum, standard deviation, median, sum, entropy, skewness, and curtosis of the observed water usage during the one-hour window.

Binary features: whether the one-hour window occurs on a Friday or not, and whether the one-hour window occurs on a Sunday or not. These two features reflect the domain knowledge about the opening hours of a store, i.e., the store has different opening hours on a Friday than the rest of the week and is closed on Sunday.

Distance to distinct consumption patterns: the Euclidean distances between the observed consumption pattern during the window and each of 20 distinct one-hour long *consumption patterns*. The latter are obtained running k-means clustering (with $k = 20$) on the union of all segmented, one-hour windows in the three stores and retrieving the cluster centroids.

Note that by using the same set of *consumption patterns* to construct the features for each store, the feature space is exactly the same for each window in each store. This satifies the critical transfer learning assumption that the source and target domain instances live in the same feature space (see Section 2), and enables transfer learning between any combination of two stores.

Table 7: The AUROC averaged over all source-target pairs of each master dataset as a function of the percentage of labels in the source domain. The table contains the results for LOCIT as well as KNNO and CORAL which are respectively the best anomaly detection and transfer learning baselines from Table 1. LOCIT's and CORAL's performance improves as labels are added to the source domain. When 10% of the source labels are available, LOCIT outperforms KNNO and CORAL on 7 of the 12 master datasets. With 50% of the source labels LOCIT always performs better or equivalently to KNNO. Compared to CORAL, LOCIT always beats it on 10 of the 12 master datasets. This table is displays the same information as in Figure 4.

Dataset	Method	Percentage of the source data that is labeled									
		10	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
Abalone	LOCIT	**0.578**	**0.575**	**0.576**	**0.576**	**0.577**	**0.576**	**0.577**	**0.578**	**0.579**	0.58
	CORAL	0.498	0.519	0.538	0.546	0.559	0.567	0.571	0.574	0.578	**0.582**
	KNNO	0.542	0.542	0.542	0.542	0.542	0.542	0.542	0.542	0.542	0.542
Covertypes	LOCIT	0.632	0.642	0.649	0.656	0.662	**0.667**	**0.672**	**0.676**	**0.68**	**0.684**
	CORAL	0.567	0.588	0.606	0.616	0.624	0.63	0.635	0.639	0.644	0.647
	KNNO	**0.663**	**0.663**	**0.663**	**0.663**	**0.663**	0.663	0.663	0.663	0.663	0.663
Gas sensor array drift	LOCIT	0.442	0.482	0.515	0.542	0.565	0.585	0.601	0.615	0.626	0.636
	CORAL	**0.628**	**0.656**	**0.665**	**0.67**	**0.676**	**0.679**	**0.684**	**0.686**	**0.689**	**0.692**
	KNNO	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565
Gesture phase segmentation	LOCIT	0.393	0.399	0.404	0.408	0.412	0.416	0.419	0.421	0.424	0.426
	CORAL	**0.491**	**0.497**	**0.505**	**0.51**	**0.516**	**0.52**	**0.524**	**0.527**	**0.528**	**0.53**
	KNNO	0.372	0.372	0.372	0.372	0.372	0.372	0.372	0.372	0.372	0.372
Recognition of handwritten digits	LOCIT	**0.985**	**0.987**	**0.989**	**0.99**	**0.99**	**0.991**	**0.991**	**0.991**	**0.991**	**0.991**
	CORAL	0.824	0.865	0.876	0.882	0.885	0.886	0.887	0.888	0.89	0.891
	KNNO	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95
Statlog landsat sattelite	LOCIT	**0.83**	**0.845**	**0.855**	**0.863**	**0.869**	**0.874**	**0.877**	**0.88**	**0.883**	**0.885**
	CORAL	0.711	0.713	0.721	0.724	0.727	0.727	0.729	0.73	0.731	0.731
	KNNO	0.799	0.799	0.799	0.799	0.799	0.799	0.799	0.799	0.799	0.799
Letter recognition	LOCIT	0.806	0.822	0.835	0.846	0.856	**0.863**	**0.871**	**0.877**	**0.882**	**0.887**
	CORAL	0.586	0.628	0.649	0.664	0.672	0.682	0.688	0.695	0.7	0.702
	KNNO	**0.86**	**0.86**	**0.86**	**0.86**	**0.86**	0.86	0.86	0.86	0.86	0.86
Pen-based recognition of digits	LOCIT	**0.935**	**0.939**	**0.943**	**0.945**	**0.947**	**0.949**	**0.951**	**0.953**	**0.954**	**0.956**
	CORAL	0.701	0.716	0.728	0.732	0.737	0.74	0.744	0.746	0.749	0.751
	KNNO	0.904	0.904	0.904	0.904	0.904	0.904	0.904	0.904	0.904	0.904
Poker	LOCIT	**0.591**	**0.59**	**0.591**	**0.59**	**0.593**	**0.595**	**0.597**	**0.6**	**0.602**	**0.604**
	CORAL	0.523	0.539	0.551	0.559	0.564	0.568	0.573	0.577	0.58	0.584
	KNNO	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54
Image segmentation	LOCIT	0.667	0.677	0.689	0.699	**0.707**	**0.714**	**0.721**	**0.727**	**0.732**	**0.736**
	CORAL	0.493	0.505	0.513	0.524	0.53	0.537	0.542	0.547	0.550	0.554
	KNNO	**0.706**	**0.706**	**0.706**	**0.706**	0.706	0.706	0.706	0.706	0.706	0.706
Shuttle	LOCIT	**0.803**	**0.816**	**0.827**	**0.836**	**0.844**	**0.851**	**0.857**	**0.862**	**0.867**	**0.871**
	CORAL	0.507	0.509	0.507	0.51	0.514	0.516	0.52	0.521	0.522	0.524
	KNNO	0.689	0.689	0.689	0.689	0.689	0.689	0.689	0.689	0.689	0.689
Waveform	LOCIT	**0.816**	**0.825**	0.831	0.836	0.84	**0.844**	**0.847**	**0.85**	**0.852**	**0.854**
	CORAL	0.805	0.823	**0.832**	**0.839**	**0.841**	**0.844**	0.844	0.845	0.848	0.847
	KNNO	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8