Conformity-Based Source Subset Selection for Instance Transfer

Neurocomputing 258 (2017) 41–51

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Conformity-based source subset selection for instance transfer

Shuang Zhou*, Evgueni Smirnov, Gijs Schoenmakers, Ralf Peeters

Department of Data Science and Knowledge Engineering, Maastricht University, P.O. Box 616, 6200 MD, Maastricht, The Netherlands

ARTICLE INFO

Article history:
Received 31 May 2016
Revised 10 October 2016
Accepted 5 November 2016
Available online 7 March 2017

Keywords:
Instance-transfer learning
Source-subset selection
Conformal test

ABSTRACT

Instance transfer aims at improving prediction models for a target domain by transferring data from related source domains. The effectiveness of instance transfer depends on the relevance of source data to the target domain. When the relevance of source data is limited, the only option is to select a subset of source data of which the relevance is acceptable. In this paper, we introduce three algorithms that perform source-subset selection prior to model training. The algorithms employ a conformity-based test that estimates the source-subset relevance based on individual instances or on subsets as a whole. Experiments conducted on four real-world data sets demonstrated the effectiveness of the proposed algorithms. Especially, it was shown that pre-training subset-selection based on set relevance is capable of outperforming the existing instance-transfer techniques.

1. Introduction

Instance-transfer learning has gained increasing attention over the last decade [1]. It aims at improving prediction models for a *target* domain by exploiting data from (closely) related *source* domains. This implies that the effectiveness of instance-transfer learning [2] depends on the relevance of the source data to the target data. Hence, estimating that relevance is essential for instance transfer [3]. However, when the relevance of the source data to the target domain is limited, the only possible solution is to select a subset from the source data which relevance to the target domain is acceptable. Thus, source-subset selection is essential for instance transfer.

Approaches to select source subsets can be described using two dimensions: (1) the order of the selection phase and model-training phase; (2) the source-relevance criterion used. For the first dimension there exist two options: pre-training selection and post-training selection. The pre-training selection first picks relevant source instances and combines them with the target data, and then trains the final model on the combined data e.g., [4]. The post-training selection however first trains a model or several models on the combination of target data and source data, and then uses these models for filtering out irrelevant source instances, e.g., [5,6]. We note that the post-training selection often requires iterating over the model-training and selection phases [5]. In this manner irrelevant source instances may introduce additional errors to the intermediate models, and eventually degrade the performance of the final ensemble. Therefore, the pre-training selection is safer than the post-training selection, especially for the source data that contains a large number of irrelevant instances.

As for the second dimension, source relevance can be determined by two criteria namely: individual relevance and set relevance. The first criterion selects a subset of source instances based on the individual relevance of each instance to the target domain [4,5]. The second criterion selects a subset of source instances based on the relevance of this subset as a whole to the target domain [6]. Later in this work we will show that the set relevance is more precise than individual relevance (see Section 5.1).

If we systematize the existing instance-transfer approaches according to the dimensions introduced above we observe that there exist: (a) algorithms with pre-training selection based on individual relevance [4]; (b) algorithms with post-training selection based on individual relevance [5,7]; and (c) algorithms with post-training selection based on set relevance [6]. However, there is no algorithm performing pre-training selection based on set relevance.Therefore, it is the aim of this paper to fill this gap. The need for designing such algorithms can be motivated from two perspectives. First, it is of interest to the systematic research in instance-transfer learning, since there is a clear methodological gap. Second, these algorithms have a potential to be promising, since as it is stated above usually the pre-training selection outperforms the post-training selection, and the set relevance is more precise than the individual relevance.

* Corresponding author.
E-mail addresses: shuang.zhou@maastrichtuniversity.nl, cattysally333@gmail.com (S. Zhou), smirnov@maastrichtuniversity.nl (E. Smirnov), gm.schoenmakers@maastrichtuniversity.nl (G. Schoenmakers), ralf.peeters@maastrichtuniversity.nl (R. Peeters).

http://dx.doi.org/10.1016/j.neucom.2016.11.071
0925-2312/

In this paper, we propose a **p**re-training **s**election algorithm based on **s**et **r**elevance (PSSR). This algorithm employs the conformal test (CT) proposed in our previous work [3] to decide whether a source subset is relevant to the target domain[1]. PSSR selects the largest source subset that passes the conformal test at a given significance level prior to the model-training phase. In case the target data is class-imbalanced, PSSR employs a class-conditional conformal test (CCCT) that proposed later in this paper. CCCT is actually the CT test that applied on target and source instances with the same class label. In consequence, the proposed PSSR algorithm can even handle class-imbalanced data, and thus is superior to the existing instance-transfer algorithms in the presence of class-imbalanced target data.

We show in this work that PSSR algorithm is effective but not computationally efficient. To address this issue we first propose a **p**re-training **s**election algorithm based on **i**ndividual **r**elevance (PSIR). This algorithm employs CT for deciding on individual relevance of any source instance prior to the model-training phase. PSIR selects the largest subset consisting of source instances that have individually passed the test. We show that identifying these instances involves a small computational cost. Thus, PSIR is a computationally efficient algorithm.

Relating the set relevance to the individual relevance, we show that CT for deciding on individual relevance can be also used for approximating CT for deciding on set relevance at a significance level of 0.5. This allows us to introduce a slight modification of the PSIR algorithm that selects a very close approximation of the largest source subset selected by PSSR at a significance level of 0.5. We call this algorithm **p**re-training **a**pproximate **s**election for 0.5-source **s**ubset (PASS).

The experimental results on real-world data demonstrate that PSSR and PASS outperform PSIR. Moreover, these two algorithms also achieve better results than four existing instance transfer algorithms that do not consider the pre-training selection and the set relevance at the same time. Thus, our main conclusion is that the combination of pre-training selection and set relevance can improve source selection for transfer, and capable of outperforming other combinations.

The remainder of this article is as follows. Section 2 provides an overview of related work. The instance-transfer task is formalized in Section 3. Section 4 describes our conformal test that decides on the relevance of a source subset to the target domain. Section 6 presents the proposed set-selection approaches. An experimental analysis is given in Section 7. Section 8 concludes the article.

2. Related work

There exist several instance-transfer algorithms that involves source-subset selection. From a prediction model view, these algorithms come from two family of ensembles: boosting ensembles [8] and bagging ensembles [9]. Below we provide an overview of these algorithms using the two dimensions for source-subset selection introduced in the previous section.

The combination of pre-training selection and individual relevance results in one algorithm from the bagging family, namely the double-bootstrapping instance-transfer algorithm [4]. This algorithm first constructs an ensemble of prediction models trained on bootstrap samples from the target data. Then the ensemble classifies the source instances and those that are correctly classified are selected. Since the selection is done through classification, the double-bootstrapping algorithm is sensitive to class-imbalanced target data. In this case the ensemble is likely to misclassify source instances of the minority class(es) even when they are in fact relevant to the target domain. Thus, the source instances from the majority class(es) are selected most of the time, and the instance transfer can become suboptimal.

The combination of post-training selection and individual relevance results in several algorithms from the boosting family, e.g., Transfer Adaboost (TrAdaBoost) [5] and Dynamic Transfer AdaBoost (Dynamic-TrAdaBoost) [7]. These algorithms are similar to the AdaBoost algorithm [8] but employ two opposite weight-update schemes depending on the type of the instances: (1) the weights of the target instances that are incorrectly classified are being increased, and (2) the weights of the source instances that are incorrectly classified are being decreased. In theory the average weighted training loss of boosting-based algorithms on the source data is guaranteed to converge to 0 as the number of iterations approaches infinity [5]. This implies that in this case relevant source instances will be classified correctly and irrelevant source instances will receive a weight of 0 (i.e., they will be totaly rejected). However, in practice, when most of the source instances are irrelevant and the size of the source data is big, these algorithms are likely to stop at very first iterations due to the fact that training error on target data in current iteration exceeds 0.5. In this case, irrelavant source instances can not be filtered out through iterations, and thus the final model is built on plenty of irrelevant source data. The latter can result in a negative transfer: when the models trained on the target and the selected source data perform worse than the models trained only on the target data.

The sensitivity of TrAdaBoost and Dynamic-TrAdaBoost to class-imbalanced target data is less obvious: it is hidden in the weight-update scheme of the source instances. Since the source instances of the minority classes have higher chance to be misclassified, they receive lower weights. Thus, they have less influence on models to be built in later iterations within the final boosting ensemble.

The combination of the post-training selection and set relevance results in one algorithm from the bagging family, namely Transfer Bagging (TrBagg) [6]. TraBagg includes two phases. In the model-training phase, first a set of bootstrap samples are randomly generated from the combined target and source data, and, then several base prediction models are trained on those samples. In the selection phase, a subset of the base prediction models are selected by minimizing the empirical error on the target data. This means that source subsets of the bootstrap samples are indirectly selected through selecting the base models and this can be viewed as hidden set relevance. Although TraBagg is simple, it requires a large number of iterations to identify all relevant source instances when the size of source data is big.

Similar to the double-bootstrapping instance-transfer algorithm, TraBagg is vulnerable to class-imbalanced data. This happens because TraBagg always selects the best prediction models with small training errors, even when such models misclassify the target instances belonging to the minority class(es).

Analyzing the instance-transfer algorithms considered so far we observe that posting-training selection is vulnerable to large and irrelevant source data, since the selection process employs information from the source data. In this respect pre-selection selection becomes more appealing. However, it may also fail in the presence of class-imbalanced target data. In addition, we note that the pre-training selection has never been combined with the set relevance which can be of interest for the systematic research in instance-transfer learning. Thus, we aim at developing a pre-training selection algorithm based on set relevance which is robust to class-imbalanced target data.

[1] The test determine the relevance of a source subset to the target domain by testing the null hypothesis that the subset is generated from the same distribution as target data under the exchangeability assumption.

3. Notations and task formalization

Let X be a feature space and Y be a class set. A domain is defined as a tuple consisting of a labeled space $(X \times Y)$ and a probability distribution P over $(X \times Y)$. We consider first a domain $\langle (X \times Y), P_T \rangle$ that we call a target domain. The target data set T is a set of m_T instances $(x_t, y_t) \in X \times Y$ drawn from the target distribution P_T under the randomness assumption (the iid assumption). Given a new test instance $x_{m_T+1} \in X$, the target classification task is to find an estimate $\hat{y} \in Y$ of the true class of x_{m_T+1} according to P_T.

Let us consider a second domain $\langle (X \times Y), P_S \rangle$ that we call a source domain. The source data set S is a set of m_S instances $(x_s, y_s) \in X \times Y$ drawn from the source distribution P_S under the randomness assumption. Knowing that the target domain and the source domain are related, we define *the instance-transfer classification task* as a classification task with an auxiliary source data set S in addition to the target data set T. We note that the class of a new test instance is estimated according to the target distribution P_T. This implies that the source data is used as auxiliary training data for the target classification task.

Instance-transfer learning is sensitive to the relevance of the source data to the target domain. Hence, estimating that relevance is essential for instance transfer. However, when the source data is not very relevant to the target domain, the only option is to select source instances which individual relevance or relevance as a set for the target domain is acceptable. Thus, the problem of selecting source instances is important for the overall success of instance transfer.

4. Conformal test to decide on instance transfer from source data

We proposed a non-parametric conformal test (CT) to decide on the relevance of source data to the target domain in [3]. The key idea is as follows. In the instance-transfer classification task, the target data and source data are generated under the randomness assumption. To decide whether the source data is relevant to the target one, we need to test a null hypothesis that the joint data set $T \cup S$ has been generated from the target distribution P_T under the randomness assumption. However, the randomness tests [10–12] are known to be incomputable [10]. Therefore, we employed the conformal prediction framework [13] to relax the randomness assumption and proposed a test under the exchangeability assumption of data generation [14][2]. Due to that assumption our test on instance transfer treats target and source data sets T and S as sequences, and it tests the null hypothesis that the concatenated data sequence TS has been generated by the target distribution P_T under the exchangeability assumption.

Below we describe our conformal test. We first introduce the p-value function used in the test and then test itself. After that, we describe a class conditional version of the test. Finally, we provide a computationally efficient approximation of the p-value function.

4.1. p-value function and test

The null hypothesis that the combined data sequence TS has been generated by the target distribution P_T under the exchangeability assumption is equivalent to the hypothesis that the probability distribution of all the permutations of the data sequence TS is uniform. Our test makes use of this equivalence and tests the latter hypothesis.

[2] The exchangeability assumption is a weaker assumption than the randomness assumption. It holds for a sequence of random variables iff the joint probability distributions of any two permutations of those variables coincide.

Vovk proposed a special case of our test for conformal prediction when the size of the source data S equals one (i.e., $m_S = 1$) in [13]. This test is based on the instance nonconformity scores as statistics for the null hypothesis. The nonconformity score $\alpha_{(x,\,y)}$ of an instance $(x, y) \in TS$ is defined as a score indicating how unusual that instance is in the data sequence $TS \backslash \{(x, y)\}$. Let $(X \times Y)^{(*)}$ represents the set of all sequences defined over $(X \times Y)$, an instance nonconformity function A is formally a mapping from $(X \times Y)^{(*)} \times (X \times Y)$ to $\mathbb{R}^+ \cup \{+\infty\}$, indicating how unusual the instance (x, y) is with respect to the instances in the data sequence $TS \backslash \{(x, y)\}$. We note that any instance nonconformity function has to produce the same result for an instance independently on the permutations of TS. Otherwise, the instance will have $|TS|!$ number of possible nonconformity scores.

Since in the instance-transfer setting the source data S usually consists of more than one instance, we generalized the work of Vovk in [3] and defined a nonconformity function for data sequences of any length. Given the combined sequence TS and any sequence $U \subseteq TS$, the nonconformity function returns a value $\alpha \in \mathbb{R}^+ \cup \{+\infty\}$ indicating how unusual the data sequence U is with respect to all the permutations with size $|U|$ of the data sequence TS.

Definition 1 (Sum sequence nonconformity function)**.** Given an instance nonconformity function A, a data sequence TS and a data sequence $U \subseteq TS$, the sum sequence nonconformity function $A^* : (X \times Y)^{(*)} \times (X \times Y)^{(*)} \to \mathbb{R}^+ \cup \{+\infty\}$ is defined as

$$A^*(TS, U) = \sum_{(x,y) \in U} \alpha_{(x,y)},$$

where $\alpha_{(x,y)} = A(T \setminus \{(x, y)\}, (x, y))$.

The sum sequence nonconformity function A^* returns the same nonconformity score for a data sequence U independently on the permutations of TS if this property holds for the instance nonconformity function A. It is also independent from the permutations of U which is important for computations.

Given an instance nonconformity function that estimates the unusualness of the instance w.r.t. the target data sequence T, we can employ the sequence nonconformity scores $\alpha_U = A^*(TS, U)$ to test the null hypothesis that the distribution of all the permutations of the data sequence TS is uniform. To design the test, we employ the p-value function defined below.

Definition 2 (p-value function)**.** Given a data sequence $U \in (X \times Y)^{(*)}$ and an integer $n \leq |U|$, the p-value function $t : (X \times Y)^{(*)} \times \mathbb{N} \to [0, 1]$ is equal to:

$$t(U, n) = \frac{|\{V \in \mathcal{P}(U, n) \mid \alpha_V \geq \alpha_{L(U,n)}\}|}{|\mathcal{P}(U, n)|},$$

where $\mathcal{P}(U, n)$ is the set of all length n permutations of U and $L(U, n)$ is the sequence of the last n elements of U.

The validity of the p-value function t has been proven in [3]. Given the combined data sequence TS and $n = m_S$, the function outputs a p-value equal to the proportion of permutations with size m_S of the sequence TS which nonconformity scores are greater than or equal to that of the source sequence S. We note that by Definition 1 the nonconformity scores employed by the t-function are computed w.r.t the target data. Hence, if we employ the initial null hypothesis, the p-value of the function t indicates the likelihood that the sequence TS has been generated by the target distribution P_T under the exchangeability assumption. The higher the p-value is, the more relevant the source sequence is to the target domain. Hence, *this p-value can be viewed as a non-symmetrical measure of relevance of the source data w.r.t. the target data.*

We employed the p-value function t in our conformal test (CT) that the combined data sequence TS has been generated by the

target distribution P_T under the exchangeability assumption. If the returned p-value is greater than or equal to a user defined significance level $\epsilon \in [0, 1]$, the null hypothesis is accepted and the entire source data S can be transferred. Otherwise, the null hypothesis is rejected together with the source set S. This gives rise to the need for source-subset selection.

4.2. Class-conditional conformal test

In the previous section, we have described our conformal test for estimating the relevance of source data to the target distribution. However, we note that when the target distribution is mainly represented by the majority class, source instances from minority class(es) tend to have very small p-values and thus hardly be selected. Therefore, in presence of class-imbalanced target data we propose to test the source data per class to see whether they follow the target distribution conditioned on that class. Since our conformal test is general enough, it is applicable for conditional distributions as well. Towards that end, we need a conditional p-value function. Given a class, the conditional p-value function is essentially our p-value function t (given in Definition 2) applied on source data from that class. The validity of this function has been proven in [15].

4.3. Approximation of the p-value function

As is stated in the previous section, the p-value function t is defined for data sequences. However, we note that the sum sequence nonconformity function A^* (as given in Definition 1) is independent from the order of the sequence U, so that the nonconformity score of a set can be defined equal to the nonconformity score of any sequence of elements of that set.

Given that the number of combinations is independent from the order of the sequence i.e., $|\mathcal{P}(S, n)| = |\mathcal{C}(S, n)| \times n!$, we re-write the p-value function definition for the case of data sets.

$$\begin{aligned} t(U, n) &= \frac{|\{V \in \mathcal{P}(U, n) | \alpha_V \geq \alpha_{L(U,n)}\}|}{|\mathcal{P}(U, n)|} \\ &= \frac{|\{V \in \mathcal{P}(U, n) | \alpha_V \geq \alpha_{L(U,n)}\}|/(n)!}{|\mathcal{P}(U, n)|/(n)!} \\ &= \frac{|\{V \in \mathcal{C}(U, n) | \alpha_V \geq \alpha_{L(U,n)}\}|}{|\mathcal{C}(U, n)|} \end{aligned} \quad (1)$$

where $\mathcal{C}(U, n)$ denotes the set of all combinations of n elements out of sequence U.

We note that the set p-value function exhibits an analogy to the notion of Wilcoxon rank-sum test (see, e.g., [16], Chapter 1). Hence, for big data sets, in which enumerating all combinations in $\mathcal{C}(TS, m_S)$ is intractable (e.g., $\mathcal{C}(100, 50)$), we propose to approximate the set p-value through Wilcoxon rank-sum test. More specifically, we assign ranks from 1 to $m_T + m_S$ to all instances in TS according to their nonconformity scores (in ascending order). In case there is a tie (i.e., instances with equal nonconformity scores), each instance in the tie is assigned a rank equal to the midpoint of unadjusted ranks in the tie span. In this setting, the nonconformity score α_V of any set V from TS with size m_S is replaced by the rank sum W that equals $\Sigma_{(x,\,y) \in V} R_{(x,\,y)}$, where $R_{(x,\,y)}$ is the rank of nonconformity score $\alpha_{(x,\,y)}$ associated with the instance $(x, y) \in V$. Accordingly, α_S is replaced by the sum of ranks of all the instances in S, denoted as W_S. In doing so, calculating $t(TS, m_S)$ reduces to calculating the probability that the rank sum of any m_S instances is bigger than that of the source instances, i.e., $P(W \geq W_S)$. To calculate $P(W \geq W_S)$ we need to know the distribution of rank sum W. Under the null hypothesis that TS is exchangeable, the sequence of ranks is also exchangeable. When the size of TS is big (i.e., m_T and m_S are bigger than 10), the rank sum W is approximately normally distributed according to the law of large number. The expectation of the rank sum is

$$E(W) = \frac{1}{2} m_S (m_T + m_S + 1)$$

and variance is

$$Var(W) = \frac{1}{12} m_T m_S (m_T + m_S + 1) - \frac{\sum_{i=1}^{e} \left(d_i^3 - d_i\right)}{(m_T + m_S)(m_T + m_S - 1)}$$

where e is the number of distinct nonconformity scores, and d_i is the number of instances at the ith tie[3]. Thus, the probability $P(W \geq W_S)$ can be easily calculated from this normal distribution. In this way we approximate the p-value $t(TS, m_S)$ with the value of $P(W \geq W_S)$.

5. Individual relevance v.s. set relevance

The proposed p-value function t is a general function. When the size of the source data set S equals one ($m_S = 1$), function t estimates the individual relevance of the single source instance in S with value $t(TS, 1)$. When the size of the source data set S is greater than one ($m_S > 1$), function t estimates the relevance of the set S as a whole with value $t(TS, m_S)$. In this section we analyze properties of individual relevance and set relevance estimated using our p-value function t.

5.1. Precision of individual relevance and set relevance

As is stated above, when $m_S = 1$ the value $t(TS, 1)$ estimates the individual relevance of a single instance in S. According to Definition 2, the value of $t(TS, 1)$ only depends on the number of target instances that are equally or more non-conformal than this source instance. Therefore, source instances with different nonconformity scores may result in the same p-value. To illustrate this finding, let us consider the following example: assume that a set TS consists of target instances t_1, t_2, t_3 followed by source instances s_1, s_2, s_3. The associated nonconformity scores are 1, 2, 6, 3, 4, 5, respectively. According to Definition 2 the individual p-values $t(T \cup \{s_1\}, 1)$, $t(T \cup \{s_2\}, 1)$, and $t(T \cup \{s_3\}, 1)$ are equal to $\frac{1+1}{3+1} = 0.5$. Hence, there is no difference among source instances s_1, s_2 and s_3 in terms of individual relevance that estimated by our p-value function t. This can be a serious problem for selection in instance transfer, since it may not be possible to distinguish source instances with different relevance.

When $m_S > 1$, the value $t(TS, m_S)$ estimates the relevance of the set S as a whole. According to Definition 2, the value of $t(TS, m_S)$ depends on the number of permutations with size m_S of the sequence TS that are equally or more non-conformal than the source data sequence S. Therefore, the set p-values are unique for sets with different nonconformity scores. Back to the example of the previous paragraph, set p-value $t(T \cup \{s_1, s_2\}, 2)$ is 0.5, set p-value $t(T \cup \{s_1, s_3\}, 2)$ is 0.4, and set p-value $t(T \cup \{s_2, s_3\}, 2)$ is 0.3. This illustrates that the set p-values are able to distinguish the sets with different nonconformity scores. Therefore, we regard the set p-value as a more precise estimation of the relevance of source data w.r.t. the target domain which is of crucial importance for selection in instance transfer.

5.2. Monotonicity of individual relevance and set relevance

Assume that the instances in source data S are sorted in increasing order of magnitude of the nonconformity scores. In this context, we analyze the monotonicity of our p-value function t w.r.t the index s of the source instances in the sorted data S. This

[3] The deviation of expectation and variance can be found in the Appendix of [16].

is done for the case of individual relevance estimation and for the case of set relevance estimation.

For the individual relevance estimation, according to Definition 2 we have that for any s: $t(T \cup \{(x_s, y_s)\}, 1) \leq t(T \cup \{(x_{s-1}, y_{s-1})\}, 1)$. Thus, in this case our p-value function t is a decreasing function of the index s, and through the index s is also a decreasing function of the nonconformity score $\alpha_{(x_s,y_s)}$.

For the set relevance estimation our analysis is more involved. Let S_s be a subset consisting of first s instances of the sorted data S. For any s we can have either $t(TS_s, s) \leq t(TS_{s-1}, s-1)$ or $t(TS_s, s) \geq t(TS_{s-1}, s-1)$. Thus, in this case our p-value function t is not a monotonic function of the index s and is not a monotonic function of the nonconformity scores $\alpha_{(x_s,y_s)}$. This result is not so obvious and that is why we provide the following example to prove our claim. Assume that TS consists of 3 target instances and 3 source instances, and corresponding nonconformity scores are {1, 4, 5, 2, 3, 6}. In this case, we have $t(TS_1, 1) = 0.75$, $t(TS_2, 2) = 0.8$ and $t(TS_3, 3) = 0.5$, which confirms the $t(TS_s, s)$ is not a monotonic function of s.

Although the p-value function t is not a monotonic function of s in general, we can still identify an interval for s where the function is indeed monotonic. Let $\alpha_{T_{max}}$ be the largest nonconformity score in the target data. By Theorem 1 (shown below) for any nonconformity score $\alpha_{(x_s,y_s)}$ that is greater than or equal to $\alpha_{T_{max}}$ we have that the p-value $t(TS_s, s)$ is smaller than or equal to the p-value $t(TS_{s-1}, s-1)$. So, if m is the index of the first source instance in the sorted data S that has a nonconformity score greater than or equal to $\alpha_{T_{max}}$, then the p-value function t is a decreasing function of the index s for the interval $[m, m_S]$.

Theorem 1. *For any instance (x_s, y_s) from the sorted source data S, if $\alpha_{(x_s,y_s)} \geq \alpha_{T_{max}}$, then $t(TS_s, s) \leq t(TS_{s-1}, s-1)$.*

Below we provide the proof of Theorem 1. Let $\alpha_D = A^*(T, D)$ denote a nonconformity score for a set D w.r.t T, S_s be defined as above, and D_s be a set of subsets D of $T \cup S_s$ whose size is s and has a nonconformity score equal to or bigger than that of S_s. Formally, $D_s = \{D \subset T \cup S_s : |D| = s, \alpha_D \geq \alpha_{S_s}\}$. We start from the following lemma that relates $|D_{s-1}|$ to $|D_s|$

Lemma 2. *If $\alpha_{(x_s,y_s)} \geq \alpha_{T_{max}}$, then $|D_s| \leq (1 + \frac{m_T}{s}) \cdot |D_{s-1}|$.*

Proof. Let D be a subset of $T \cup S_{s-1}$ with $|D| = s-1$. Now we consider the set $T \cup S_s$ and we will add one element of $(T \cup S_s) \backslash D$ to D to create a set E of size s. We distinguish between two cases:

1. $D \notin D_{s-1}$ or $\alpha_D < \alpha_{S_{s-1}}$. There are $m_T + 1$ ways to create the set $E \subset T \cup S_s$. For all of these sets we have

   $$\alpha_E \leq \alpha_D + \alpha_{(x_s,y_s)} < \alpha_{S_{s-1}} + \alpha_{(x_s,y_s)} = \alpha_{S_s}$$

   and hence $E \notin D_s$.
2. $D \in D_{s-1}$ or $\alpha_D \geq \alpha_{S_{s-1}}$. Again, there are $m_T + 1$ ways to create the set $E \subset T \cup S_s$. One way to create E is to add (x_s, y_s) to D. This gives $E = D \cup \{(x_s, y_s)\}$, and in this case

   $$\alpha_E = \alpha_D + \alpha_{(x_s,y_s)} \geq \alpha_{S_{s-1}} + \alpha_{(x_s,y_s)} = \alpha_{S_s}$$

   and hence $E \in D_s$.
   The other m_T ways to create E are by adding one of the m_T elements of $(T \cup S_{s-1}) \backslash D$ to D, thus there are in total $m_T \cdot |D_{s-1}|$ number of E's. However, the resulting E will be created as a superset of in total s sets of size $s-1$. For example, a set $E = \{(x_1, y_1), (x_2, y_2), (x_3, y_3)\}$ is a superset of $D = \{(x_1, y_1), (x_2, y_2)\}$, $D = \{(x_1, y_1), (x_3, y_3)\}$, and $D = \{(x_2, y_2), (x_3, y_3)\}$. In this case, E will be created three times from different Ds. So, assuming that the newly created set E satisfies $\alpha_E \geq \alpha_{S_s}$, which is not necessarily the case, it will be created s times and it should of course count only once towards $|D_s|$. In other words, there are at most $\frac{m_T}{s} \cdot |D_{s-1}|$ number of distinct E such that $\alpha_E \geq \alpha_{S_s}$.

Combining the two results, we find: $|D_s| \leq (1 + \frac{m_T}{s}) \cdot |D_{s-1}|$, which completes the proof. □

By using Lemma 2, we prove Theorem 1 as follows:

Proof. Let D_s be a set of subsets D of $T \cup S_s$ which size is s and has a nonconformity score equal to or bigger than that of S_s, and assume that $\alpha_{(x_s,y_s)} \geq \alpha_{T_{max}}$. We have

$$\begin{aligned}
t(TS_s, s) &= \frac{|\{D \in \mathcal{C}(TS_s, s) | \alpha_D \geq \alpha_{S_s}\}|}{|\mathcal{C}(TS_s, s)|} && \text{by Eq. (1)} \\
&= \frac{|D_s|}{\binom{m_T+s}{m_T}} \\
&\leq \frac{\left(1 + \frac{m_T}{s}\right)|D_{s-1}|}{\binom{m_T+s}{m_T}} && \text{by Lemma 2} \\
&= \frac{\left(1 + \frac{m_T}{s}\right)}{\binom{m_T+s}{m_T}} \cdot \binom{m_T + s - 1}{m_T} \cdot t(TS_{s-1}, s-1) \\
&= \frac{\left(1 + \frac{m_T}{s}\right)s}{m_T + s} \cdot t(TS_{s-1}, s-1) \\
&= t(TS_{s-1}, s-1)
\end{aligned}$$

□

6. Set selection

In this section, we introduce two algorithms for source-subset selection: a **p**re-training **s**election algorithm based on **s**et **r**elevance (PSSR) and a **p**re-training **s**election algorithm based on **i**ndividual **r**elevance (PSIR). We show that PSSR is more precise for selection while the PSIR is more computationally efficient . To balance between these two algorithms, we introduce a slight modification of PSIR namely **p**re-training **a**pproximate **s**election for 0.5-source **s**ubset (PASS). We show that this algorithm is computationally efficient and precise for selection at a significance level of 0.5.

6.1. Pre-training selection algorithm based on set relevance

The pre-training selection algorithm based on set relevance (PSSR) is given in Algorithm 1. Given a target data set T, a source data set S, significance level ϵ, and an instance nonconformity function A, it outputs the *largest* source subset $S^* \subseteq S$ that passes the test at the significance level ϵ. The computation process is implemented as follows. The algorithm first computes the nonconformity scores for all the target and source instances using the instance nonconformity function A. More precisely, the nonconformity score $\alpha_{(x_t,y_t)}$ for any target instance $(x_t, y_t) \in T$ is calculated w.r.t. subset $T \backslash \{(x_t, y_t)\}$, and the nonconformity score $\alpha_{(x_s,y_s)}$ for any source instance $(x_s, y_s) \in S$ is calculated w.r.t. T. Then, it determines the maximal target nonconformity score $\alpha_{T_{max}}$ and sorts the source instances in increasing order of the nonconformity scores. After that, the algorithm initializes the largest source subset S^* by including all the source instances $(x_s, y_s) \in S$ with nonconformity scores $\alpha_{(x_s,y_s)}$ smaller than $\alpha_{T_{max}}$. By Theorem 1 the p-value function t is a decreasing function for the nonconformity scores greater than or equal to $\alpha_{T_{max}}$. Therefore, if the set p-value $t(TS^*, |S^*|)$ of the subset S^* is bigger than or equal to ϵ, the p-value of any superset of S^* decreases as more source instances added. This allows the algorithm to apply the binary-search method to find the final largest source subset S^* that is subsequently output.

If the set p-value $t(TS^*, |S^*|)$ of the subset S^* is smaller than ϵ, then the algorithm reduces minimally S^*. It does by sequentially removing instances from the sorted S^* starting with the instance with the highest nonconformity score until the p-value of S^* equals or exceeds ϵ. We note that this process is sequential due to the fact

Algorithm 1 Pre-training selection algorithm based on set relevance.

Input: Target data T, Source data S, Significance level ϵ, Instance nonconformity function A.
Output: The largest subset $S^* \subseteq S$ s.t. $t(TS^*, |S^*|) \geq \epsilon$.

1: **for** each target instance $(x_t, y_t) \in T$ **do**
2: Set the nonconformity score $\alpha_{(x_t,y_t)}$ equal to $A(T \setminus \{(x_t, y_t)\}, (x_t, y_t))$.
3: **end for**
4: **for** each source instance $(x_s, y_s) \in S$ **do**
5: Set the nonconformity score $\alpha_{(x_s,y_s)}$ equal to $A(T, (x_s, y_s))$.
6: **end for**
7: Determine the maximal target nonconformity score $\alpha_{T_{max}}$.
8: Sort the source data S in increasing order of the nonconformity scores $\alpha_{(x_s,y_s)}$.
9: Set the largest set S^* equal to $\{(x_s, y_s) \in S : \alpha_{(x_s,y_s)} < \alpha_{T_{max}}\}$.
10: **if** $t(TS^*, |S^*|) \geq \epsilon$ **then**
11: Set the left counter L equal to $|S^*|$ and the right counter R equal to $m_S - 1$.
12: **while** $L \leq R$ **do**
13: Set the middle index m equal to $\lfloor \frac{L+R}{2} \rfloor$.
14: Set p-value p_m of set S_m equal to $t(TS_m, |S_m|)$.
15: Set p-value p_{m+1} of set S_{m+1} equal to $t(TS_{m+1}, |S_{m+1}|)$.
16: **if** $p_m \geq \epsilon$ and $p_{m+1} < \epsilon$ **then**
17: Set the set S^* equal to the set S_m.
18: **break**.
19: **else if** $p_s > \epsilon$ **then**
20: Set $L = m + 1$.
21: **else**
22: Set $R = m - 1$.
23: **end if**
24: **end while**
25: **else**
26: **while** $t(TS^*, |S^*|) < \epsilon$ **do**
27: Exclude the last instance from S^*.
28: **end while**
29: **end if**
30: **output** S^*.

that the p-value function t is not monotonic for the nonconformity scores smaller than the maximal target conformity score $\alpha_{T_{max}}$.

PSSR is computationally inefficient because of the sequential application of the p-value function t in the worst case (see line 26 in Algorithm 1). Nevertheless, PSSR provides more precise set selection due to the fact that set p-values are a more precise estimation of the relevance of source data w.r.t. the target domain.

6.2. Pre-training selection algorithm based on individual relevance

The pre-training selection algorithm based on individual relevance (PSIR) is given in Algorithm 2. Given a target data set T, a source data set S, significance level ϵ, and an instance nonconformity function A, it selects the *largest* source subset $S^* \subseteq S$ such that any instance $(x_s, y_s) \in S^*$ individually passes the test at ϵ. The selection process is implemented as follows. First, the algorithm initializes the largest subset S^* as an empty set. The nonconformity scores for the target and source instances are calculated by the instance nonconformity function A similarly to the previous algorithm. After that, the source instances are sorted in increasing order of the nonconformity scores $\alpha_{(x_s,y_s)}$. Since the p-value function t is a decreasing function of the nonconformity scores $\alpha_{(x_s,y_s)}$, the individual p-values of source instances are arranged in decreasing order of magnitude. This allows the algorithm to apply the binary-search method to find the last source instance with a p-value greater than or equal to the significance level ϵ. This instance is the right border of the *largest* source subset S^* in the sorted data S. Therefore, all the source instances from S with indices smaller than or equal to m are added to the final subset S^* that is subsequently output by the algorithm.

Algorithm 2 Pre-training selection algorithm based on individual relevance.

Input: Target data T, Source data S, Significance level ϵ, Instance nonconformity function A.
Output: Largest subset $S^* \subseteq S$ s.t. $\forall (x_s, y_s) \in S^*$, $t(T \cup \{(x_s, y_s)\}, 1) \geq \epsilon$.

1: Set the largest set S^* equal to $\emptyset$.
2: **for** each target instance $(x_t, y_t) \in T$ **do**
3: Set the nonconformity score $\alpha_{(x_t,y_t)}$ equal to $A(T \setminus \{(x_t, y_t)\}, (x_t, y_t))$.
4: **end for**
5: **for** each source instance $(x_s, y_s) \in S$ **do**
6: Set the nonconformity score $\alpha_{(x_s,y_s)}$ equal to $A(T, (x_s, y_s))$.
7: **end for**
8: Sort the source data S in increasing order of the nonconformity scores $\alpha_{(x_s,y_s)}$.
9: Set the left counter L equal to 1 and the right counter R equal to $m_S - 1$[1].
10: **while** $L \leq R$ **do**
11: Set the middle index m equal to $\lfloor \frac{L+R}{2} \rfloor$.
12: Set p-value p_m of instance $(x_m, y_m) \in S$ equal to $t(T \cup \{(x_m, y_m)\}, 1)$.
13: Set p-value p_{m+1} of instance $(x_{m+1}, y_{m+1}) \in S$ equal to $t(T \cup \{(x_{m+1}, y_{m+1})\}, 1)$.
14: **if** $p_m \geq \epsilon$ and $p_{m+1} < \epsilon$ **then**
15: Add source instances from S with indices smaller than or equal to m to S^*.
16: **break**.
17: **else if** $p_m > \epsilon$ **then**
18: Set $L = m + 1$.
19: **else**
20: Set $R = m - 1$.
21: **end if**
22: **end while**
23: **output** S^*.

PSIR is computationally efficient because of the binary-search method used. However, due to the fact that source instances with quite different nonconformity scores may result in the same individual p-value, the source subset is not precise.

6.3. Pre-training approximate selection for 0.5-source subset

If a source subset is generated by the target distribution, the expected p-value of this subset is equal to 0.5. This implies that this subset is randomly drawn from the target distribution and, thus, it is can be transferred. We call such a subset as *0.5-source subset* $S^{0.5}$ and show below an efficient way to select a source subset that is approximately $S^{0.5}$.

So far we have demonstrated that using PSSR for the 0.5-source subset $S^{0.5}$ (i.e., the largest source subset S^* with set p-value greater than or equal to 0.5) is computationally inefficient. However, a very precise approximation $\hat{S}^{0.5}$ of the subset $S^{0.5}$ can be computed at a small cost. Assume that the source data S is sorted in increasing order of the nonconformity scores $\alpha_{(x_s,y_s)}$ and S_n is a subset consists of the first n instances of the ordered source data S. By Theorem 3 (see below) if the average of individual p-values of all instances in a source subset S_n equals $0.5 + \frac{1}{2(m_T+1)}$, then the

set p-value of S_n is approximately 0.5. This means that the subset S_n is an approximation $\hat{S}^{0.5}$ of the 0.5-source subset $S^{0.5}$.

The pre-training approximate selection for 0.5-source subset (PASS) is very similar to PSIR given in Algorithm 2. It employs the fact that the average of individual p-values is decreasing with the nonconformity scores of instances in sorted source data S due to the monotonicity of the p-value function t of those scores. Therefore, the binary-search method is applied again to efficiently generate the largest source subset S^* which in this case equals the approximate subset $\hat{S}^{0.5}$. This implies the 0.5-source set selection algorithm differs from the set-selection shown in Algorithm 2 in line 14. Instead of testing the individual p-value p_m at the middle point, the algorithm tests the average p-value of all the source instances until the middle point.

Theorem 3. *If the average of individual p-values of all the instances in the subset $S_n \subset S$ is equal to $0.5 + \frac{1}{2(m_T+1)}$, then the set p-value of S_n is approximately 0.5.*

Proof. According to Definition 2, for any source instance $(x_s, y_s) \in S_n$, the number of target instances that have nonconformity scores greater than or equal to that of (x_s, y_s) is equal to $p_s * (m_T + 1) - 1$, where p_s is the individual p-value of (x_s, y_s). Thus, the number of target instances that associated with smaller nonconformity scores is $m_T - p_s * (m_T + 1) + 1$.

Since all the instances in S_n are sorted in increasing order of nonconformity scores, there are $0, 1, \ldots, n-1$ source instances with smaller nonconformity scores than $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, respectively. That is to say for any $(x_s, y_s) \in S_n$ there are $s-1$ source instances with smaller nonconformity scores than that of (x_s, y_s).

Combining these two parts, there are in total $m_T - p_s * (m_T + 1) + 1 + (s-1)$ instances ranked higher than (x_s, y_s). Assuming that there is no tie, the rank of (x_s, y_s) is:

$$R_s = m_T - p_s * (m_T + 1) + 1 + s \tag{2}$$

Therefore the rank sum of all instances in S_n is:

$$\begin{aligned} R_{S_n} &= \sum_{s=1}^{n} (m_T - p_s * (m_T + 1) + 1 + s) \\ &= n(m_T + 1) - (m_T + 1) \sum_{s=1}^{n} p_s + \sum_{s=1}^{n} s \\ &= n(m_T + 1) - (m_T + 1) n \left(\frac{1}{2} + \frac{1}{2(m_T + 1)} \right) + \frac{1}{2} n(n+1) \\ &= n \left(m_T + 1 - \frac{1}{2}(m_T + 1) - \frac{1}{2} + \frac{1}{2} n + \frac{1}{2} \right) \\ &= \frac{1}{2} n (m_T + n + 1) \end{aligned} \tag{3}$$

According to the Wilcoxon Ranksum test, the expectation of rank sum of any source subset of size n is equal to $\frac{1}{2}n(m_T + n + 1)$. Since the rank sum of S_n is equal to the expectation, the set p-value of S_n is approximately 0.5. □

We prove Theorem 3 under the assumption that there are no ties in the nonconformity scores. If there are some source instances tied together, midpoint of this tie span is used as the rank. In this case the rank sum of these source instances stays the same, and thus the p-value of the subset S_n is still approximately 0.5. In case there are some source instance tied with target instances, the rank sum gets bigger than that given in Eq. 3. Assume that we have d_t target instances and d_s source instances attached on one tie, and there are in total l instances ranked higher than all instances in this tie. Then the ranks of source instances in this tie are equal to $l + \frac{1}{2}(1 + d_t + d_s)$, instead of $l+1, l+2, l+d_s$. Therefore, the difference in rank sum caused by this tie will be:

$$d_s * \left(l + \frac{1}{2}(1 + d_t + d_s) \right) - \sum_{i=1}^{d_s} (l + i) = \frac{1}{2} d_s d_t$$

In case of ties' existence, the rank sum will increase by $\sum_{i=1}^{e} \frac{1}{2} d_{s_i} d_{t_i}$, where e is the number of ties. Thus, the set p-value will be lower than 0.5. To limit the number of such ties a proper nonconformity function needs to be chosen (an injective function in the best case).

By Theorem 3 the source subset S_n is the approximation set $\hat{S}^{0.5}$, if the average of individual p-values of the instances in S_n equals $0.5 + \frac{1}{2(m_T+1)}$. In fact, it is self-evident that for a target set of a reasonable size the term $\frac{1}{2(m_T+1)}$ can be ignored. That is to say for any source subset, if the average individual p-values of all instances in this subset is 0.5, then its set p-value is approximately 0.5, and thus it is the approximate 0.5-source set $\hat{S}^{0.5}$.

7. Experiments

This section presents our experimental results and conclusions. We first provide the experiment setup in Section 7.1. Then, we present instance-transfer tasks under study in Section 7.2. Finally, in Section 7.3, we evaluate the generalization performance of proposed algorithms and compare them with existing instance-transfer techniques.

7.1. Experiment setup

To set up our set-selection algorithms we needed to set up the instance nonconformity function employed in the p-value function t. This setup was done depending on the dimensionality of the data. In our experiments we had two types of data: non-text data with relatively low dimensionality and text data with high dimensionality. For the non-text data we used the nearest-neighbor instance nonconformity function [10]. The nearest neighbor instance nonconformity function A_{NN} outputs for the target data T and an instance (x_i, y_i) a nonconformity score $\frac{\sum_{j=1}^{k} d_{ij}^{+}}{\sum_{j=1}^{k} d_{ij}^{-}}$, where k is number of neighbors, d_{ij}^{+} is the distance from x_i to the jth closest instance in T having the same class label as x_i, and d_{ij}^{-} is the distance from x_i to the jth closest instance in T having a different class label. For the text data, due to the high dimensionality, we used the general instance nonconformity function defined in [13]. The general nonconformity function A_G outputs for the target data T and an instance (x_i, y_i) a nonconformity score $\sum_{y \in \mathbf{Y}, y \neq y_i} s_y$, where s_y is the score of class $y \in Y$ produced by a classifier trained on target data T for the instance x_i. In our experiments we employed Random Forest [17] as a non-conformal classifier.

We noticed that in most of the instance-transfer tasks the target data are class-imbalanced, and thus we employed the conditional p-value function to estimate the relevance of source data.

The significance level ϵ for PSIR was set equal to 0.5. This is due to the fact that if a source instance is randomly drawn from the target distribution, its p-value is expected to be greater than or equal to 0.5. Analogously, the significance level ϵ for PSSR was set equal to 0.5. That is because if a source subset is a random sample from the target distribution, its p-value is expected to be equal to 0.5 as well. The significance level ϵ for PASS was 0.5 by default.

Having selected source subset, we applied Support Vector Machines (SVM) [18] with linear kernel to train a prediction model on the combination of target data and selected source data. WEKA's[19] implementation of SVM with default setting was used in the experiments.

Table 1
Landmine detection instance-transfer classification tasks.

Datasets		Description	Size	p-value
Landmine	T	Instances from Mine 26 to 29	1799	1.0
	S1	Instances from Mine 1 to 5	3086	0.17
	S2	Instances from Mine 6 to 10	2547	0.27
	S3	Instances from Mine 11 to 15	2902	0.24
	S4	Instances from Mine 16 to 20	2240	0.47
	S5	Instances from Mine 21 to 25	2246	0.45

To assess the quality of instance transfer we employed the generalization performance of the instance-transfer classifier represented by the Area Under the ROC Curve (AUC). The method of AUC estimation was a stratified holdout method on the target data repeated 100 times. For the non-text data (text data), 10% (4%) of instances were randomly sampled from the target data for training and the remaining for testing. The smaller percentage for the text data was due to the fact that for bigger percentage instance transfer is no longer required.

The set-selection algorithms were compared with four instance-transfer algorithms presented in Section 2: TrAdaBoost, Dynamic-TrAdaBoost, TraBagg, and DoubleBootStrap. All the algorithms used SVM (with linear kernel) as a base classifier. The number of iterations was set to 100 for all the four algorithms.

7.2. Instance-transfer classification tasks

Four real-world data sets were used in our experiments. They are described below:

- Landmine detection[4] is a collection of 29 data sets related to detecting landmine in 29 different landmine fields. The 29 data sets have different distributions due to different geographic conditions. For example, data sets 1 to 15 correspond to foliated regions while sets 16 to 29 correspond to regions that have bare earth. In this context we derived target and source data sets as follows. Data sets 26 to 29 were combined together and used as the target data set. Data sets 16 to 20 and 21 to 25 were combined into two source data sets with a high similarity to the target one while data sets 1 to 5, 6 to 10, and 11 to 15 were combined into other three source data sets with a lower similarity. The target data set and a source data set defined together one instance-transfer classification task. For each task, 10% of instances were randomly sampled from the target set for training and the remaining for testing. The p-values of the relevance of the source data to the target data (computed by the p-value function t) are given in the last column of Table 1.
- Wine quality[5] is a data set of in 1599 red-wine and 4898 white-wine instances. Each instance is represented by 11 physiochemical features (e.g., PH values) and a grade given by expert. In the experiments, red-wine instances were used as the target data set and five source data sets were sampled from white-wine instances based on different conditions. The target data set and a source data set defined together one instance-transfer classification task. The p-values for the source data sets of all the tasks are given in the last column of Table 2.
- 20-Newsgroups[1] is a data set of about 20,000 news documents organized in a two-level hierarchy. The hierarchy consists of 7 top categories and 20 subcategories. For example, 'comp' and 'sci' are two top categories such that 'comp' has two subcategories, 'comp1' and 'comp2', and 'sci' has two subcategories, 'sci1' and 'sci2'. Five instance-transfer classification tasks were defined as top-category tasks such that the target and source data were drawn from different subcategories. For each task 50 instances were randomly sampled from the target data for training and the remaining for testing. The p-values of the source data are given in the last column of Table 3.
- Reuters-21578 [1] is a collection of data sets with text documents organized in hierarchical structures. Three instance-transfer classification tasks were defined in the same way as those of the 20-newsgroups task. For each task 50 instances were randomly sampled from the target data for training and the remaining for testing. The p-values of the target relevance of the source data are given in the last column of Table 4.

7.3. Experimental results

The generalization performance (AUCs) for PSIR, PSSR and PASS are given in Tables 5–8. In addition, the generalization performance (AUCs) of four instance-transfer algorithms mentioned in Section 2, TrAdaBoost, Dynamic-TrAdaBoost, TraBagg, and DoubleBootStrap are given for comparison. Since all the algorithms in our experiments used SVM as a base classifier, the generalization performance (AUC) of SVM on the target data (case of no instance transfer) is given as a baseline classifier. The maximal AUC for each row is given in bold. A series of paired t-tests was performed at the significance level of 0.05 to compare the performance of instance-transfer classifiers to that of the baseline classifier. The statistically significant improvement is marked with "+", while the statistically significant negative transfer is marked with "−". A second series of paired t-tests was performed at the significance level of 0.05 to compare the AUCs of PSIR, PSSR and PASS to those of TrAdaBoost, Dynamic-TrAdaBoost, TraBagg, and DoubleBootStrap. If PSIR (PSSR, PASS) is statistically better than all the four algorithms, it is marked with *.

From the Tables 5 to 8 we see that PSSR has the best generalization performance over all the instance-transfer classification tasks. It achieves the highest AUC in 12 out of 18 tasks. PASS performs comparably well as PSSR in most of the tasks, since it is a close approximation of PSSR at a significance level of 0.5. PSSR slightly outperforms PASS in some of the tasks due to the fact that in the existence of ties the p-values of the sets returned by PASS are a bit lower than 0.5. However, PSSR involves much bigger computational cost comparing to PASS due to the sequential application of the p-value function. In Table 9, we compare the average time that PSSR and PASS spend to search for the 0.5-source set from sorted source data. As shown in the table, the computation time of PSSR may even be 10 times more than that of PASS in the search step, which confirms its computational inefficiency.

PSIR has the second best generalization performance (achieves the highest AUC in 3 out of 18 tasks). It is slightly worse than PSSR and PASS in particular for the tasks where the relevance of the source data to the target domain is relatively high. The reason is twofold. First, the selection based on set relevance is more precise than the selection based on individual relevance (as explained in Section 5.1). Second, PSIR is more conservative: it selects a subset of the source instances selected by PSSR and PASS. To clarify, we give an example for the "orgs vs people" task. We sorted the instances from the source data in increasing order of the nonconformity scores. Then, we added the source instance with the lowest nonconformity score to a preliminary empty source subset and computed the set p-value . We repeated the last step till all the source instances were added, and we plotted the obtained subset p-values against the size of the source subsets in Fig. 1a. After that, we repeated the same process for individual relevance, i.e., instead of computing subset p-values we computed individual p-value for the last instance of each subset, and then we plotted the individual p-values in Fig. 1a.

[4] http://www.cse.ust.hk/TL/
[5] https://archive.ics.uci.edu/ml/datasets/Wine+Quality

Table 2
Wine quality instance-transfer classification tasks.

Datasets		Description	Size	*p*-value
Wine	T	Red wine	1599	
	S1	White wine that *total sulphur dioxide* > 0.2 and 0.16 < *acid* < 0.22	1548	0.2
	S2	Random sample of white wine	1469	0.21
	S3	White wine that *alcohol* < 0.5 and 0.16 < *acid* < 0.22	1499	0.24
	S4	White wine that *total sulphur dioxide* > 0.2	1499	0.29
	S5	White wine that *alcohol* < 0.5	1540	0.33

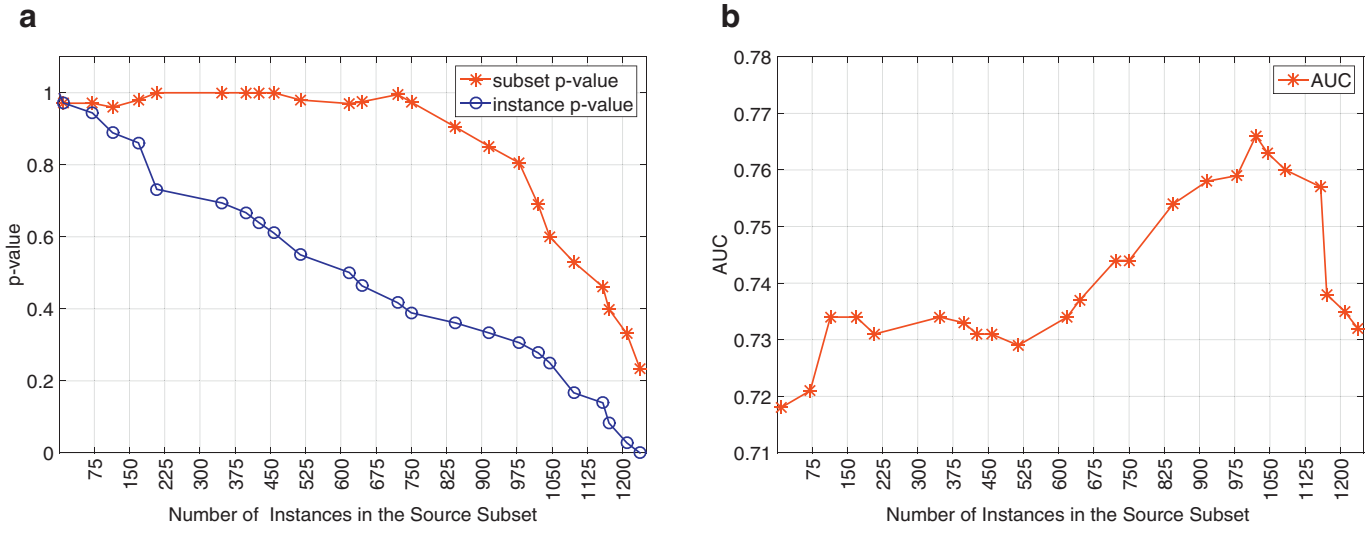

Fig. 1. (a): Individual *p*-values and subset *p*-values w.r.t. source instances ordered by the nonconformity scores. (b): AUCs of SVM trained on the target data and growing subsets of source instances ordered by the nonconformity scores.

Table 3
20-Newsgroups instance-transfer classification tasks.

Datasets	Tasks	Sample size		*p*-value
		$\|T\|$	$\|S\|$	
20-Newsgroups	Comp vs sci	3930	4900	0.30
	Rec vs talk	3669	3561	0.32
	Rec vs sci	3961	3965	0.34
	Sci vs talk	3374	3828	0.34
	Comp vs talk	4482	3652	0.39

Table 4
Reuters-21578 instance-transfer classification tasks.

Datasets	Tasks	Sample size		*p*-value
		$\|T\|$	$\|S\|$	
Reuters	People vs places	1239	1210	0.15
	Orgs vs places	1079	1080	0.27
	Orgs vs people	1016	1046	0.37

Comparing the plots of set *p*-values and individual *p*-values, we observe that the set *p*-value is much bigger than the individual *p*-value for the same subset. At the significance level of 0.5 PSIR selects the first 645 instances from the sorted source data, while PSSR and PASS select the first 1083 and 1158 instances from the sorted source data, respectively. That is why PSIR exhibits a more conservative selection. As a result, it benefits less from the relevant source data. Fig. 1b presents the generalization performance of a prediction model (SVM) trained on the target data and growing subsets of the same sequence of the sorted source instances. The maximization of the model performance happens for the subset contains 1020 instances from the sorted source data, which is very close to the set selected by PSSR. On the contrary, PSIR stops much earlier with limited benefit from instance transfer. The subset selected by PASS contains more irrelevant source instances than the 0.5-source set, and results in a set *p*-value equals to 0.46. That explains why PASS gets close but a bit lower AUC than PSSR.

From the Tables 5 to 8 we see that the PSIR, PSSR and PASS outperform TrAdaBoost, Dynamic-TrAdaBoost, TraBagg and DoubleBootStrap. They are significantly better than the other four algorithms in more than half of the tasks. For example, for the 20-newsgroups tasks PSSR and PASS are significantly better than other algorithms with a margin of 0.1 (see Table 7). For the Reuters-21578 tasks PSSR, PASS, and PSIR result in positive transfers: they have improved the generalization performance comparing to the base line classifier. However, TrAdaBoost, Dynamic-TrAdaBoost and TraBagg result in a negative transfer, especially for the "people-vs-places" task. Our algorithms achieve better results due to following

Table 5
Performance of instance-transfer transfer algorithms for the landmine tasks.

Datasets	Source	*p*-value	Baseline	PSIR	PSSR	PASS	TrAdaBoost	Dynamic TrAdaBoost	TraBagg	Double Bootstrap
Landmine	S1	0.17	0.53	0.54	0.54	0.54	0.54	$\mathbf{0.55}^{+}$	0.53	0.53
	S2	0.27	0.53	$\mathbf{0.55}^{+}$	0.54	0.54	$\mathbf{0.55}^{+}$	0.54	0.53	0.53
	S3	0.24	0.53	$0.57^{+}*$	$\mathbf{0.58}^{+}*$	0.56^{+}	0.55^{+}	0.54	0.53	0.53
	S4	0.47	0.53	0.61^{+}	$\mathbf{0.63}^{+}*$	$\mathbf{0.63}^{+}*$	0.60^{+}	0.60^{+}	0.58^{+}	0.59^{+}
	S5	0.45	0.53	0.61^{+}	$\mathbf{0.63}^{+}*$	$\mathbf{0.63}^{+}*$	0.58^{+}	0.59^{+}	0.56^{+}	0.59^{+}

Table 6
Performance of instance-transfer transfer algorithms for the wine tasks.

Datasets	Source	p-value	Baseline	PSIR	PSSR	PASS	TrAdaBoost	Dynamic TrAdaBoost	TraBagg	Double-Bootstrap
Wine	S1	0.20	0.52	**0.66**$^{+}*$	0.65^{+}	0.64^{+}	0.62^{+}	0.64^{+}	0.55^{+}	0.55^{+}
	S2	0.21	0.52	**0.67**$^{+}*$	0.66$^{+}*$	0.66$^{+}*$	0.63^{+}	0.64^{+}	0.57^{+}	0.55^{+}
	S3	0.24	0.52	**0.67**$^{+}*$	0.66^{+}	0.66^{+}	0.65^{+}	0.66^{+}	0.57^{+}	0.55^{+}
	S4	0.29	0.52	0.67^{+}	**0.68**$^{+}$	**0.68**$^{+}$	0.66^{+}	**0.68**$^{+}$	0.59^{+}	0.56^{+}
	S5	0.33	0.52	0.67^{+}	**0.69**$^{+}$	0.68^{+}	0.68^{+}	0.67^{+}	0.59^{+}	0.57^{+}

Table 7
Performance of instance-transfer transfer algorithms for the 20-Newsgroups tasks.

Datasets	Source	p-value	Baseline	SSIR	PSSR	PASS	TrAdaBoost	Dynamic TrAdaBoost	TraBagg	Double-Bootstrap
20News -groups	Comp vs sci	0.30	0.51	0.59$^{+}*$	**0.64**$^{+}*$	0.61$^{+}*$	0.54^{+}	0.53	0.53	0.52
	Rec vs talk	0.32	0.51	0.64$^{+}*$	**0.68**$^{+}*$	**0.68**$^{+}*$	0.61^{+}	0.62^{+}	0.58^{+}	0.53
	Rec vs sci	0.34	0.52	0.64^{+}	**0.66**$^{+}*$	**0.66**$^{+}*$	0.64^{+}	0.64^{+}	0.65^{+}	0.53
	Sci vs talk	0.34	0.51	0.68$^{+}*$	**0.72**$^{+}*$	**0.72**$^{+}*$	0.64^{+}	0.65^{+}	0.62^{+}	0.53
	Comp vs talk	0.39	0.50	0.72$^{+}*$	**0.76**$^{+}*$	0.74$^{+}*$	0.66^{+}	0.68^{+}	0.64^{+}	0.53

Table 8
Performance of instance-transfer transfer algorithms for the Reuters-21578 tasks.

Datasets	Source	p-value	Baseline	SSIR	PSSR	PASS	TrAdaBoost	Dynamic TrAdaBoost	TraBagg	Double-Bootstrap
Reuters 21578	People vs places	0.15	0.70	**0.73**$^{+}*$	0.72$^{+}*$	0.72$^{+}*$	0.56^{-}	0.56^{-}	0.62^{-}	0.69
	Orgs vs places	0.27	0.70	0.73$^{+}*$	**0.75**$^{+}*$	0.73^{+}	0.71	0.69^{-}	0.72^{+}	0.71
	Orgs vs people	0.37	0.72	0.74^{+}	**0.76**$^{+}$	**0.75**$^{+}$	**0.73**$^{+}$	**0.74**$^{+}$	**0.76**$^{+}$	**0.76**$^{+}$

Table 9
Average search time for 0.5-source sets of PSSR and PASS, recorded in milliseconds.

Datasets	Average search time (ms)	
	PSSR	PASS
Landmine	11.02	0.63
Wine	4.41	0.61
20News-groups	3.65	0.51
Reuters-21578	1.28	0.53

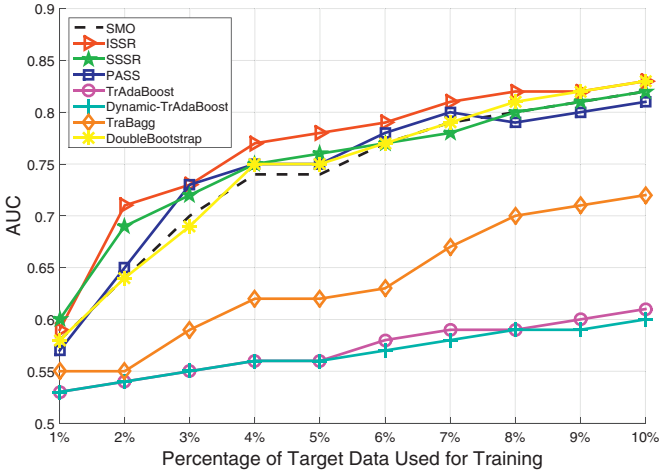

Fig. 2. The AUCs as functions of the size of training data for the "people vs places" task.

two reasons. First, our algorithms employ pre-training selection of source instances. In contrast, TrAdaBoost, Dynamic-TrAdaBoost and TraBagg employ post-training selection. As is stated in Section 2, post-training algorithms can not effectively filter out irrelevant source instances when the algorithms stop at early iterations (e.g., because of big training errors). As a result, they suffer from negative transfer when the relevance of source data is limited. Secondly, our algorithms are robust against class-imbalanced target data thanks to the usage of the class-conditional p-value function. On the contrary, the other four algorithms mostly select instances from majority class(es) in the presence of class-imbalanced target data. This limits the performance of the final prediction model.

In Fig. 2 we focus on the "people vs places" task, where the relevance of the source data to the target data is low. The percentage of target data used for training is gradually increased from 1% to 10%. AUCs of all classifiers are plotted against the size of the target training data. It can been seen from the figure that pre-training instance-transfer classifiers, PSIR, PSSR, PASS and DoubleBootstrap, never result in negative transfer. They improve the model when the training size is very small (e.g., less than 5%). On the contrary, post-training instance-transfer algorithms are very vulnerable to irrelevant source data. This observation confirms our claim that pre-training selection is superior to post-training selection. PSIR achieves the best results for this task due to the conservative selection it employs. We believe that it is a safer choice for source data with a low p-value (e.g. less than 0.2). If we generalize the experimental results, we conclude that instance-transfer algorithms that employ pre-training selection for source instances are better than those that employ post-training selection. They achieve promising results in the whole range of source data from relevant to less relevant w.r.t. the target domain. PSIR, PSSR and PASS outperform existing post-training source-selection algorithms because of the statistical soundness and the ability of handling class-imbalanced target data. Comparing PSSR and PASS to PSIR, we conclude that by applying pre-training selection based on set relevance, the final model can benefit more from instance transfer.

8. Conclusion

In this paper, we introduced three pre-training source-subset selection algorithms, PSSR, PSIR, and PASS, for instance transfer. The algorithms are statistically sound due to the conformal test (CT) employed and robust against class-imbalanced target data due to the class-conditional version of the same test. PSIR estimates the source subset relevance using individual instance relevance and it is computationally efficient. PSSR estimates the source subset relevance using set relevance and it is computationally inefficient. PASS

is essentially PSIR that approximates PSSR at the significance level of 0.5 and it is proposed as a computationally efficient substitute of PSSR.

Experiments demonstrated that PSSR, PSIR and PASS outperform existing post-training and pre-training transfer algorithms. In addition they showed that PSSR and PASS outperform PSIR. Thus, we may conclude that pre-training selection with set relevance is a superior approach for instance selection in the context of instance transfer.

References

[1] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.
[2] L. Torrey, J. Shavlik, Transfer learning, Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques 1 (2009) 242.
[3] S. Zhou, E. Smirnov, G. Schoenmakers, R. Peeters, K. Driessens, A non-parametric conformity-based test for transfer decisions, in: Proceedings of the IEEE 27th International Conference on Tools with Artificial Intelligence, IEEE, 2015, pp. 628–635.
[4] D. Lin, X. An, J. Zhang, Double-bootstrapping source data selection for instance-based transfer learning, Pattern Recognit. Lett. 34 (11) (2013) 1279–1285.
[5] W. Dai, Q. Yang, G.-R. Xue, Y. Yu, Boosting for transfer learning, in: Proceedings of the 24th International Conference on Machine Learning, ACM, 2007, pp. 193–200.
[6] T. Kamishima, M. Hamasaki, S. Akaho, Trbagg: a simple transfer learning method and application to personalization in collaborative tagging, in: Proceedings of the 9th IEEE International Conference on Data Mining, 2009, pp. 219–228.
[7] S. Al-Stouhi, C.K. Reddy, Adaptive boosting for transfer learning using dynamic updates, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2011, pp. 60–75.
[8] R.E. Schapire, The strength of weak learnability, Mach. Learn. 5 (2) (1990) 197–227.
[9] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[10] V. Vovk, A. Gammerman, G. Shafer, Algorithmic Learning in A random World, Springer, 2005.
[11] P. Martin-Löf, The definition of random sequences, Inf. Control 9 (6) (1966) 602–619.
[12] A. Church, On the concept of a random sequence, Bull. Am. Math. Soc. 46 (2) (1940) 130–135.
[13] G. Shafer, V. Vovk, A tutorial on conformal prediction, J. Mach. Learn. Res. 9 (2008) 371–421.
[14] D.J. Aldous, Exchangeability and related topics, in: École d'été de probabilités de Saint-Flour, XIII—1983, 1117, Springer, 1985, pp. 1–198.
[15] V. Vovk, Conditional validity of inductive conformal predictors, J. Mach. Learn. Res. - Proc. Track 25 (2012) 475–490.
[16] E.L. Lehmann, H.J. D'Abrera, Nonparametrics: Statistical Methods based on Ranks, Springer, New York, 2006.
[17] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[18] C. Cortes, V. Vapnik, Support vector machine, Mach. Learn. 20 (3) (1995) 273–297.
[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The WEKA data mining software: an update, SIGKDD Explor. 11 (2009) 304–339.

Shuang Zhou received a bachelors degree in software engineering from University of Electronic Science and Technology of China, UESTC, in 2009. She received a masters degree in artificial intelligence from Department of Data Science and Knowledge Engineering (DKE), Maastricht University, the Netherlands, in 2012, and continued her study in DKE as a PhD candidate from 2012 till now. Her research interests include transfer learning, conformal prediction, and their applications.

Evgueni Smirnov is an assistant professor of Artificial Intelligence at the Department of Knowledge Engineering, Maastricht University. He graduated in Computer Science from the Technical University of Sofia in 1988, and he earned his PhD degree in Artificial Intelligence at Maastricht University in 2001. Research interests include: Data mining (reliable prediction, feature selection); Machine learning (transfer learning, ensemble learning, kernel methods, version spaces); Applications of data mining to medicine, transportation, machinery-automation systems, and education. Evgueni Smirnov has co-edited several books on reliable data mining (Springer, IEEE). He supervised/executed six commercial data-mining projects. His team of PhD students does research on transfer learning, ensemble learning, and medical data mining.

My name is **Gijs Schoenmakers**. I am Assistant Professor at the Department of Knowledge Engineering (DKE) at Maastricht University, The Netherlands. My main research field is Game Theory. Within this field my research focuses primarily on equilibria and equilibrium refinements in repeated and stochastic games. The most important result I achieved in this field is establishing the existence of subgame perfect equilibria in recursive perfect information games (joint work with Jeroen Kuipers, Janos Flesch and Koos Vrieze). Last year I was asked to join DKEs research on Machine Learning. My task here primarily consists of uncovering mathematical structures within the concepts that are being researched.

Ralf Peeters (1964) is a full professor in Applied Mathematics at Maastricht University. He graduated at Delft University of Technology (1988) and received his Ph.D. degree from the Free University, Amsterdam (1994). He currently is vice-chair and Research Director of the Department of Data Science and Knowledge Engineering. His research interests include: system identification and machine learning, signal processing, data science, optimization, and applications of knowledge engineering to medicine and the life sciences.