Expert Systems With Applications 115 (2019) 565–577

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Seeded transfer learning for regression problems with deep learning

Syed Moshfeq Salaken[a,b,*], Abbas Khosravi[a], Thanh Nguyen[a], Saeid Nahavandi[a]

[a] Institute of Intelligent Systems Research and Innovation, Deakin University, Australia
[b] Deakin University, 75 Pigdons Road, Waurn Ponds, Victoria 3216, Australia

ARTICLE INFO

Article history:
Received 25 May 2018
Revised 24 August 2018
Accepted 24 August 2018
Available online 27 August 2018

Keywords:
Transfer learning
Domain adaptation

ABSTRACT

The difference in data distributions among related, but different domains is a long standing problem for knowledge adaptation. A new method to transform the source domain knowledge to fit the target domain is proposed in this work. The proposed method uses deep learning method and limited number of samples from target domain to transform the source domain dataset. It treats the limited samples of target domain as seeds for initiating the transfer of source knowledge. Comprehensive experiments are conducted using different computational intelligence models and different datasets. Obtained results reveal that prediction models trained using the proposed method demonstrate the best performance in comparison with the same models trained with only source knowledge or deep learned features. Experiments show that models trained using proposed method have outperformed the baseline methods by at least 50% in 14 experiments out of a total of 18.

© 2018 Elsevier Ltd. All rights reserved.

1. Introduction

Human ability to learn a new concept using different but related topic is very fascinating. Current machine learning (ML) algorithms designed by researchers are not able to achieve a similar success because conventional ML methods are dictated by the underlying assumption of same probability distribution in training and testing sets (Dai, Yang, Xue, & Yu, 2007; Pan & Yang, 2010). In real life, this assumption often gets violated and ML algorithms need retraining using relevant data. This is a costly process in terms of money and time (Pan & Yang, 2010). At the same time, abundance of data is available from a different, but somehow related, domain. This fact motivated researchers to undertake efforts to utilize these apparently useless data to build a better model in the domain of interest (Calais Guerra, Veloso, Meira Jr, & Almeida, 2011; Dai, Yang et al., 2007; Pan & Yang, 2010).

Presently, transfer learning has shown its worth in a wide variety of contexts including web document classification (Aghamaleki & Baharlou, 2018; Al-Mubaid, Umair et al., 2006; Dai, Xue, Yang, & Yu, 2007b; Fung, Yu, Lu, & Yu, 2006; Han, Liu, & Fan, 2018; Sarinnapakorn & Kubat, 2007), medical image processing (Khatami et al., 2018), brain-computer interface calibration (Hossain, Khosravi, Hettiarachchi, & Nahavandi, 2018; Hossain, Khosravi, & Nahavandi, 2016), sentiment classification in social media (Ghiassi & Lee, 2018; Glorot, Bordes, & Bengio, 2011; Pang, Lee, & Vaithyanathan, 2002; Tan, Wu, Tang, & Cheng, 2007), wifi user location estimation (Pan, Zheng, Yang, & Hu, 2008) and product review (Afrin, Nepal, & Monplaisir, 2018; Calais Guerra et al., 2011). Most of the researchers in this field have approached the research question from a classification point of view while regression problems are largely ignored (Dai, Yang et al., 2007). However, transfer learning is equally important in regression contexts as well because of its potential to provide better prediction in a difficult environment. For example, Pardoe et. al (Dai, Yang et al., 2007) approached the transfer learning problem from a regression point of view and showed prediction accuracy can be greatly improved in target domain using knowledge gathered from different domains. They have used multiple source domains for enhanced prediction in target domain using modified version of two existing classification transfer algorithms, namely ExpBoost and TrAdaBoost, with boost and stack techniques. Our work differs from this method because, we do not use any boosting method or stacking method and we consider only one source domain in our work. In addition, to the best of our knowledge, no work exists till date that combines the power of deep network and adaptation to transfer the knowledge in target domain in a regression context.

In this work, we address this gap in research and transfer learning for regression context is tackled with a limited availability of target domain data and deep learning mechanism. It is shown that adaptation of deep learned and more abstract representation of source domain can yield higher prediction success in target do-

* Corresponding author.
E-mail addresses: syed.salaken@deakin.edu.au (S.M. Salaken), abbas.khosravi@deakin.edu.au (A. Khosravi), thanh.nguyen@deakin.edu.au (T. Nguyen), saeid.nahavandi@deakin.edu.au (S. Nahavandi).

https://doi.org/10.1016/j.eswa.2018.08.041
0957-4174/© 2018 Elsevier Ltd. All rights reserved.

main. The proposed method is not specific to any training algorithm/methodology because it produces a transformed and more effective feature set to train with. It is important to note that, proposed method alters the output of source domain during transformation process as well and therefore, it is essentially a supervised learning.

In summary, the contribution of this paper is a new algorithm that adapts the samples from source domain to approximately match the target domain. The adaptation is done not in original feature space, but rather in abstracted feature space which is derived through the use of deep feature learning methods.

The rest of the paper is organized as follows: Section 2 discusses current results in transfer learning domain, Section 3 describes the proposed knowledge transfer mechanism, Section 4 describes the dataset used in this work, Section 5 describes the experimental set-up, Section 6 discusses the results and finally, Section 7 concludes the article.

2. Related studies

According to Pan and Yang, transfer learning is defined as a method of helping the learning process of the target predictive function $f_T(.)$ to predict learning task τ_T in target domain D_T by utilizing the knowledge in source domain D_S, target domain D_T and source predictive learning task τ_S under the assumption that $D_S \neq D_T$, or $\tau_S \neq \tau_T$ (Pan & Yang, 2010). From probabilistic point of view, a domain can be considered as pair $D = \{X, P(X)\}$ where $P(X)$ defines the probability distribution of the feature space. Similarly, a task can be considered a pair defined as $T = \{Y, P(Y|X)\}$ where Y is the label space with probability distribution $P(Y|X)$. Most importantly, the definition in Pan and Yang (2010) implies that one of the following is true: $X_S \neq X_T$ or $P_S(X) \neq P_T(X)$, $Y_S \neq Y_T$ or $P(Y_T|X_T) \neq P(Y_S|X_S)$ (Pan & Yang, 2010). Our work follows this definition by ensuring the distribution of label space (e.g. output space, which is a continuous variable for regression problem) is different in source and target domain in real datasets. For a synthetic dataset used in this study, distributions in both feature space and output space are different between source and target domain to ensure the proposed method works in more challenging case as well. A description of the datasets used in this work is given in Section 4. Strictly speaking, this work is very similar to multi-task learning (a subset of inductive transfer learning as per Pan & Yang, 2010) because labeled data is available in both source and target domains. However, because source and target tasks are not learned at the same time, it is not the same as multi-task learning. Although the examples in this work are performing same task in both source and target domain, the method is not limited to that assumption. Therefore, this work is applicable to both inductive and transductive learning under the condition that labeled data is available in both domains.

Considering different approaches to transfer learning, four different families of methods exist till date for knowledge transfer (Pan & Yang, 2010). Instance based methods perform by re-weighting some source data to be used in target domain (as training data) (Bickel, Brückner, & Scheffer, 2007; Dai et al., 2007b; Dai, Yang et al., 2007; Fan, Davidson, Zadrozny, & Yu, 2005; Huang, Gretton, Borgwardt, Schölkopf, & Smola, 2006; Jiang & Zhai, 2007; Liao, Xue, & Carin, 2005; Quionero-Candela, Sugiyama, Schwaighofer, & Lawrence, 2009; Sugiyama, Nakajima, Kashima, Buenau, & Kawanabe, 2008; Zadrozny, 2004). Feature based approaches work mainly by finding some common features that essentially contributes to better result in target domain (Ando & Zhang, 2005; Argyriou, Evgeniou, & Pontil, 2008; Argyriou, Pontil, Ying, & Micchelli, 2007; Blitzer, Dredze, Pereira et al., 2007; Blitzer, McDonald, & Pereira, 2006; Dai, Xue, Yang, & Yu, 2007a; Daumé III, 2009; Evgeniou & Pontil, 2007; Jebara, 2004; Lee, Chatalbashev, Vickrey, & Koller, 2007; Raina, Battle, Lee, Packer, & Ng, 2007; Wang & Mahadevan, 2008). Parameter based transfer methods work by digging out shared parameters or priors in both domains (Bonilla, Chai, & Williams, 2007; Evgeniou & Pontil, 2004; Gao, Fan, Jiang, & Han, 2008; Lawrence & Platt, 2004; Schwaighofer, Tresp, & Yu, 2004) and relational knowledge transfer models work by building a mapping between different domains (Davis & Domingos, 2009; Mihalkova, Huynh, & Mooney, 2007; Mihalkova & Mooney, 2008; Pan & Yang, 2010). However, the proposed method in this work first finds some abstract representation of source domain via deep learning techniques, and then applies a transfer mechanism to modify the derived abstract representation to fit the target domain. In other words, it tries to synthetically generate more target domain data in places where more target domain samples are likely to be found. Later, these data are used in training phase of any traditional ML algorithm.

From a different perspective, transfer learning algorithms can be divided into approaches that use models trained on source data and models that use source data directly (Dai, Yang et al., 2007c). Our work follows the approach where model is trained by using source data. In addition, our work can also be considered similar to domain adaptation approach which is a subset of transfer learning (Pan & Yang, 2010). There are methods that can act as baseline for the domain adaptation based transfer learning approaches. Possible baselines are 1) source only baseline, which ignores the target data and trains a single model only on source data, 2) target only models that ignore source data and train a single model on target domain data only and finally, 3) ALL baseline that trains a model by considering both source and target domain data (Daumé III, 2009). This work assumes that not enough data is available in the target domain to train a model. Therefore, source only model is treated as baseline in this work. The ALL baseline is ignored because our approach can be used in scenario where source and target domain predictors can be different, both in characteristics and numbers. For example, source domain features can be time, light, temperature (3 features) while target domain features can be velocity and force (2 features). In such a case, ALL baseline cannot be used because source and target domain data cannot be treated as one unified data source. Only restriction on our algorithm is that number of features after deep learning phase must be equal to number of features in target domain. Deep learning technique used as a pre-processing step provides us with the following advantages over other possible feature engineering steps:

- deep learning is proven to find underlying structure in the data. Therefore, it should be helpful to transform the underlying structure from one domain to another because it, theoretically, provides a defence against noise and erroneous samples.
- a source domain may have less features than the target domain. In these cases, other feature engineering methods can provide an increased number of features to satisfy the requirement of our proposed method (i.e. “number of features after deep learning phase must be equal to number of features in target domain”) without the hope to reveal a hidden structure. Techniques like PCA cannot be used to reveal underlying pattern in these cases at all. However, deep learning can easily handle both feature reduction or inflation scenario.

There are only a few studies in transfer learning that utilize deep learning (see Gharehbaghi & Lindén, 2017 for a related study). Davis et al. used second order Markov logic to discover structural regularities in source domain (Davis & Domingos, 2009). This method is inspired by analogical reasoning (Davis & Domingos, 2009; Falkenhainer, Forbus, & Gentner, 1989). They have considered three different transfer techniques of the discovered structural regularities, one of which uses seeding the beam method. It involves deep transfer by seeding, as described by Davis et al., “the

Table 1
Distinctions of our method with current literature.

Current literature	Our approach
Uses second-order Markov Logic to capture structural regularities using predicates and variables and transfer them by replacing predicate names (Davis & Domingos, 2009)	Learns structural regularities via deep features and transfers via seeding techniques
Uses deep learning methods to learn abstract features (Bengio, 2012; Mesnil et al., 2012)	Uses deep learning methods to learn abstract features
Uses target domain inputs to learn variations in target domain (Mesnil et al., 2012)	Uses target domain samples to transform learned deep features to fit the target domain
Single or muliple sources are used with boosting to transfer knowledge (Dai, Yang et al., 2007)	Only one source domain is used while transfer is done by modifying learned features

beam with the clauses obtained from each legal instantiation of a clique in the target domain" (Davis & Domingos, 2009). Our work utilizes a similar concept of seeding where available labeled target domain samples are used to instantiate the transfer mechanism. Bengio et al. (Bengio, 2012; Mesnil et al., 2012) used deep learning to learn representation in unsupervised and transfer learning scenario for classification task. They intended to learn abstract features in the higher levels of representation. This is probably motivated by the work of Raina et al. (2007) where the author learned a big number of abstract features from unrelated data using sparse coding and deep learning. Our work shares the same belief as well. However, none of them used target domain data to transform the source domain samples.

A brief comparison between our method and current literature (only those which address either regression problem using transfer learning or use deep learning for knowledge transfer) is given in Table 1.

Next, a brief description of three learning methods used in this work is provided for completeness of this work.

- Adaptive Network based Fuzzy Inference System (ANFIS): ANFIS (Jang, 1993) is a hybrid adaptive network based learning framework which is functionally equivalent to Takagi-Sugeno-Kang (TSK) (Takagi & Sugeno, 1985) fuzzy inference framework. It uses adaptive network topology to learn non-linear mapping among input(s) and output(s) from a number of observations/samples.

  A fuzzy inference system is one that uses fuzzy sets (Zadeh, 1965) to capture uncertainties of human reasoning by assigning a membership degree to each input value. A very straightforward example of fuzzy set and membership degree can be given by asking what temperature inside a room can be considered as somewhat cold, cold and very cold. The answer will vary and depend on a number of factors. However, it is indeed possible to define a range of temperature that defines a very cold condition. Based on opinions from a number of people, it is then possible to determine which value of temperature most people consider as very cold. Therefore, that particular temperature can be given a certain degree of belongingness to very cold temperature. This degree of belongingness is called membership degree in fuzzy terminology and is bounded between 0 and 1. Often a parameterized function is used to determine the membership degree and therefore it is called membership function. In ANFIS, parameters of membership functions are considered premise parameters in a fuzzy inference framework. Coupled with membership degree and a rulebase in the form of "IF input-01 is x and input-02 is y, THEN output is z", a fuzzy system can determine the effect of a new input vector on its rulebase and can determine the corresponding output by aggregating the outputs of all rules. In a TSK system, $z = ax + by + c$, where a, b and c are called consequent parameters. The structure of fuzzy set helps to handle uncertainty from the beginning of the design and therefore fuzzy sets have been proven effective in dealing with uncertainty as well as imprecise and noisy data (Mendel, 2007).

  ANFIS learns the parameter of fuzzy systems from sample observations instead of expert or intuitive knowledge. The topology of ANFIS contains 5 layers. The first layer is a fuzzification layer where each of the inputs are assigned a membership value. The second layer is multiplication layer that produces the functionality of product t-norm in fuzzy inference system and provides firing strength of all rules. The third layer computes ratio of firing strength of all rules. In the fourth layer, output of each rule is calculated using consequent parameters for individual rules along with the firing strength. Finally, the fifth layer takes a summation of the output of all rules to produce the final output. In the learning scheme, a forward pass till layer 4 attempts to approximate consequent parameters using least square method and a backward pass attempts to approximate premise parameters using gradient descent method.
- Interval type-2 fuzzy logic system (IT2FLS): The fuzzy sets discussed above have only one membership degree for each value of an input. For example, if 5 °C is considered a member of cold temperature set then it has a membership degree within this set. Assume the membership degree is 0.6. Therefore, this particular temperature has only one membership degree in each set it belongs to. As a result, it does not have any uncertainties in the belongingness to cold set anymore. This type of fuzzy sets are called type-1 fuzzy set (T1FS). However, if there is a situation that prevents us from assigning a fixed membership degree to the 5 °C, we can solve the problem by assigning a continuous interval of membership degree to cold set for this temperature. In such a situation, this fuzzy set will be called an interval type-2 fuzzy set (IT2FS). Any fuzzy system that utilizes an IT2FS is called interval type-2 fuzzy logic system (IT2FLS). The inference procedure for IT2FLS is similar to type-1 fuzzy logic system (T1FLS) except for the fact that an additional step (called type reduction) is needed after the inference process to produce a crisp output. IT2FS provides an additional degree of freedom in the design process of a fuzzy system and therefore it has capability of handling uncertainty in a more powerful way. the parameters of an IT2FLS are often determined through an optimization process such as genetic algorithm. For more details, the reader is encouraged to read (Liang & Mendel, 1999; 2000).
- Artificial Neural Network (ANN): ANN is a powerful gradient descent based learning method that is designed as a network of interconnected nodes (Haykin & Network, 2004). ANN has at least 3 layers, namely input layer, hidden layer(s) and output layer. It is possible to have multiple hidden layers. Each layer contains several nodes, also called neurons. Each layer may also contain a bias node. The number of nodes in the first (i.e. input) layer is defined by the number of inputs in the system. Number of hidden layer neurons are variable and the optimal number needs to be found by an optimization process. Output layer neurons are limited to the number of outputs of the system. Each neuron contains a processing functionality. For input layer, this functionality is often achieved by activation functions. These functions usually take an input and pro-

duce an output which is normally bounded between either [0,1] or $[-1, 1]$. In the hidden layer, each neuron takes a weighted sum of outputs from the input layer and then fires an activation function. Similar chain of events takes place between subsequent layers. The learning mechanism follows a gradient descent method where both forward and backward passes are used to tune the weights in the neural network (Demuth, Beale, De Jess, & Hagan, 2014; Haykin & Network, 2004).

3. Knowledge transfer mechanism - seeding technique

This work assumes that there are actual outputs available in target domain. In addition, it is assumed that the available target domain data is representative of complete target domain. However, the amount of available data is not sufficient to train a learner.

Moreover, as this is an investigation in transfer learning, abundance of source domain data is a prerequisite. The aim of transfer learning method is to learn some knowledge from these source domain data and utilize them to increase prediction accuracy in target domain.

In other words, the goal of this research is to find a method which will allow the use of limited target domain data in addition with source domain samples in order to increase prediction accuracy for new samples in target domain. One way of achieving this goal is to adapt source domain according to available target domain samples.

Here, we employ a simple seeding technique for this adaptation. Each of the available data samples in target domain is considered a seed for our purpose. Next, a clustering is performed on the source domain samples. Number of clusters is derived from the number of target domain samples. Therefore, if there are N_T samples available from the target domain, a N_T point clustering will be performed on the source domain samples. In this work, k-means clustering technique is utilized for this purpose. However, any clustering method can be used instead of k-means. The algorithm does not specifically need k-means algorithms. It is used for the sake of simplicity in this work.

Finally, source domain clusters are assigned to the target domain sample points considering them as new cluster centers. The assignment is done by looking at minimal Euclidean distance between source domain cluster centroid and corresponding target domain samples. This process also involves adapting source domain samples based on the Euclidean difference of cluster members and cluster centroid in feature space. This effectively transfers clusters from source domain to target domain while preserving source domain structure. Note that, this method does not demand equal number of source and target domain features. The only requirement is to ensure that number of deep features, which is derived prior to the application of the proposed transfer mechanism, is equal to the number of target domain features. This research utilized autoencoder to generate the abstract, deep features. However, other deep learning methods can be used to generate them as well.

By doing so, following key points are achieved:

- Source domain knowledge is preserved in the form of cluster structure e.g. shape in the transferred domain.
- The value ranges of source domain features are translated to those of target domain features
- More target domain samples are synthetically produced in areas where more actual target domain data is likely to be found. This is a way of predicting future target domain samples under the assumption that available target domain samples are fully representative of complete target domain.

For the purpose of providing a concise example on how the algorithm works, let's consider an original source domain dataset with 12 samples and 5 features i.e. $D_{SO}^{12\times5}$ and a target domain dataset with 4 samples and 3 features i.e. $D_T^{4\times3}$. The proposed algorithm requires the source domain to be abstracted via an encoder to produce an abstracted source domain $D_S^{12\times3}$ to match the number of features in target domain. Based on the dimension of source and target domain, we may need to deflate or inflate the source domain feature space. This is done before the algorithm can be applied. By convention, if the number of features in source and target domain matches, we still perform a feature abstraction to get deep representation of source domain features. The learning process inside the encoders is described in experimental set-up section. Once this abstraction is completed, $D_S^{12\times3}$ is passed to the seeded knowledge transfer algorithm. At this stage, $D_S^{12\times3}$ is clustered using a k-means algorithm where number of clusters is 4. Within each of these clusters, their center is calculated. Afterwards, correspondence of these 4 clusters are established with 4 available target domain samples by utilizing minimum Euclidean distance. If there is a tie, it's broken at random. Once this clusters are associated with target domain samples, distance between individual cluster member and cluster center is calculated and stored in a corresponding adjusting matrix, $M_A^{aa\times3}(i); i = 1, 2, \ldots, 4$. Here, aa is the number of members in each cluster. Finally, each M_A is added to corresponding target sample (after replicating individual target sample aa times) and vertically concatenated to produce adapted dataset $D_A^{12\times3}$. At this point, any conventional learning algorithm (regression or classification) can be applied to D_A.

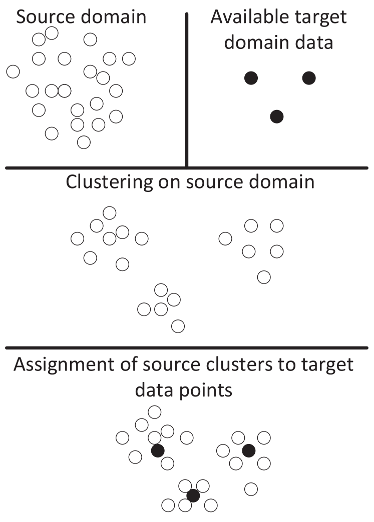

Source domain
Available target domain data
Clustering on source domain
Assignment of source clusters to target data points

Fig. 1. Knowledge adaptation method.

A visual representation of this process is described in Fig. 1. Step by step description is given in Algorithm 1.

4. Datasets

All of the learning tasks involved in this experiment is regression problem i.e. prediction of continuous variable. We have not considered any categorical variables in either input or output side throughout this task. The aim of this task is to make available a small sample of data from target domain in the learning phase, transfer knowledge from source domain using proposed algorithm and test on remaining dataset once the learning part is completed. The source domain is adapted based on the proposed algorithm and 3 different learning methods are applied on each of the following 3 datasets.

Algorithm 1 Seeded knowledge transfer algorithm.

```
Input: $D_S^{n_S \times m_S}$(deep features from source domain, derived using
    autoencoder),
    $D_T^{n_T \times m_T}$(available target domain samples) where $m_S = m_T$. $D_S$
    and $D_T$ contains output in the last column and each row represents an observation. Here, $D$, $S$, $T$, $n$, $m$ denotes dataset, source
    domain, target domain, number of samples and number of features, respectively.
Output: $D_A^{n_A \times m_T}$ (adapted/transformed dataset, to be used in model
    training).
    Initialization :
1:  $n_A$ = $n_S$, number of samples in $D_A$
    empty matrix $D_A^{n_A \times m_T}$
    Clustering :
2:  Call k-means algorithm on $D_S$ with $n_T$ as "number of clusters"
    parameter
    Transfer:
3:  For each cluster, find difference of cluster center and members
    of cluster in the form of Euclidean distance.
4:  Store the difference for each cluster in an adjusting matrix,
    $M_A^{m_{aa} \times m_T}(i); i = 1, 2, \ldots, n_T$. $aa =$ number of members in cluster.
5:  Find one-to-one correspondence between each cluster center
    and $D_T(i)(i = 1, 2, \ldots n_T)$ based on the lowest Euclidean distance between them. In the events of a tie, it is broken at random.
6:  Add each $M_A$ with corresponding $D_T$. Replicate each $D_T^{1 \times m_T}$ to
    match the dimension of $M_A$. Store the result in $Q_A$ e.g. $Q_A(i) =$
    $D_T(i) + M_A(i), i = 1, 2, \ldots, n_T$
7:  Vertically concatenate all $Q_A(i)$ and call it $D_A$
8:  return $D_A$
```

4.1. Synthetic data 01

This synthetic data is generated for an imaginary two-input, one-output system. Source domain inputs follow uniform distribution and bound between 0 and 10. Output in the source domain is defined as per the below equation:

$$O_{S1} = a + b + a * b \tag{1}$$

where a and b are inputs in the system. Target domain inputs follow a Gaussian/Normal distribution and are bound between 0 and 5. Output in this domain is defined as per the below equation:

$$O_{T1} = a + \frac{b^2 + a}{b^3} \tag{2}$$

4.2. Intel laboratory data

This is a dataset collected in the Intel Berkeley Research Laboratory using Mica2Dot sensors with weatherboard which contains timestamp, humidity, light, temperature and voltage information. Data is collected every 31 seconds using TinyDB in-network query processing set-up, which is built on TinyOS framework. The data contains information collected from 54 sensors, which are located in different parts of laboratory, between February 28, and April 5, 2004. For the purpose of this research, method used by J. Shell et al. (Shell & Coupland, 2015) is followed to separate source and target domain data. This ensures that both temporal and spatial shifts occur between source and target domain data. More specifically, source data is collected from sensor 21 on February 28, 2004 and target data is collected from sensor 46 on March 5, 2004.

A two-input, one-output system is formed from this source and target domain data. Input variable contains light and time, which is converted in seconds with milliseconds portion removed, while temperature is used as output. This follows the procedure described in Shell and Coupland (2015). A pictorial view of output for both source and target domain data in Intel laboratory dataset is shown in Fig. 2a and Fig. 2b.

4.3. Townsville wireless sensor network

This dataset is a collection of sensor data from a trial project of wireless sensor network deployment by researchers in James Cook University, Townsville, Australia (Mohring, Myers, Atkinson, VanDerWal, & van der Valk, 2015; Mohring, 2014). The sensors are placed in 11 different sensor platforms in Townsville municipal office building. Date, timestamp, air temperature, surface temperature, case temperature, humidity, light, noise and voltage are collected between July 14, 2014 to 31 July, 2014. Source data is constructed from 4 sensors (unit 9–12) and target domain is constructed from sensor 8. There are 7 inputs and 1 output in the system. Temperatures, humidity, light and noise are considered as inputs and battery voltage is considered as output in this system. A pictorial view of output in both source and target domain is shown in Fig. 2c and d respectively. These figures indicate that there are substantial differences between variable distributions in the source and target domain.

5. Experimental set-up

During this work, one synthetic dataset and two real datasets are used to verify the effectiveness of proposed method. Synthetic method facilitates controlled experimentation and challenges the proposed algorithm in extreme fashion. No other dataset in this work has different distribution and range in all features and outputs for both source and target domain. For the two real datasets, one (Intel lab dataset) is taken from an indoor environment, while the other (Townsville wireless network dataset) was taken from outdoor environment. For the Intel lab data, output is preprocessed using a log-transformation while a number of constant features (e.g. current) are removed from Townsville wireless network dataset.

Once the datasets are chosen and processed, they are converted into matrices. Columns contain features while rows represent samples. Next, higher level abstractions of source domain features are computed using auto-encoders. In this case two autoencoders are used to derive the deep features. Output of the first auto-encoder is fed as input into the second one. Second auto-encoder produces final feature set which has the same number of features as the target domain data. Each of these autoencoders is composed of one encoder and one decoder. Encoder unit inside the autoencoder maps the input vector to another vector by processing through a transfer function, a weight matrix and a bias vector. The decoder unit maps the encoded vector to the abstract representation of the original input. The learning process of each autoencoder involves a standard sparsity-constrained L2-regularized cost function minimization problem as below:

$$L(\theta) = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K}(x_{kn} - \hat{x}_{kn})^2 + \Omega_w + \beta * \Omega_\rho \tag{3}$$

where x represents the input to the autoencoder, $\hat{x}$ represents the abstract representation, N is the number of observations, K is the number of features, Ω_w is the $L2$ regularization term, β is sparsity coefficient and Ω_ρ is the sparsity regularization term. Ω_ρ can be defined as per the following equation.

$$\Omega_\rho = \sum_{i=1}^{m}\left(\rho \log\left(\frac{\rho}{\hat{\rho}_i}\right) + (1-\rho)\log\left(\frac{1-\rho}{1-\hat{\rho}_i}\right)\right) \tag{4}$$

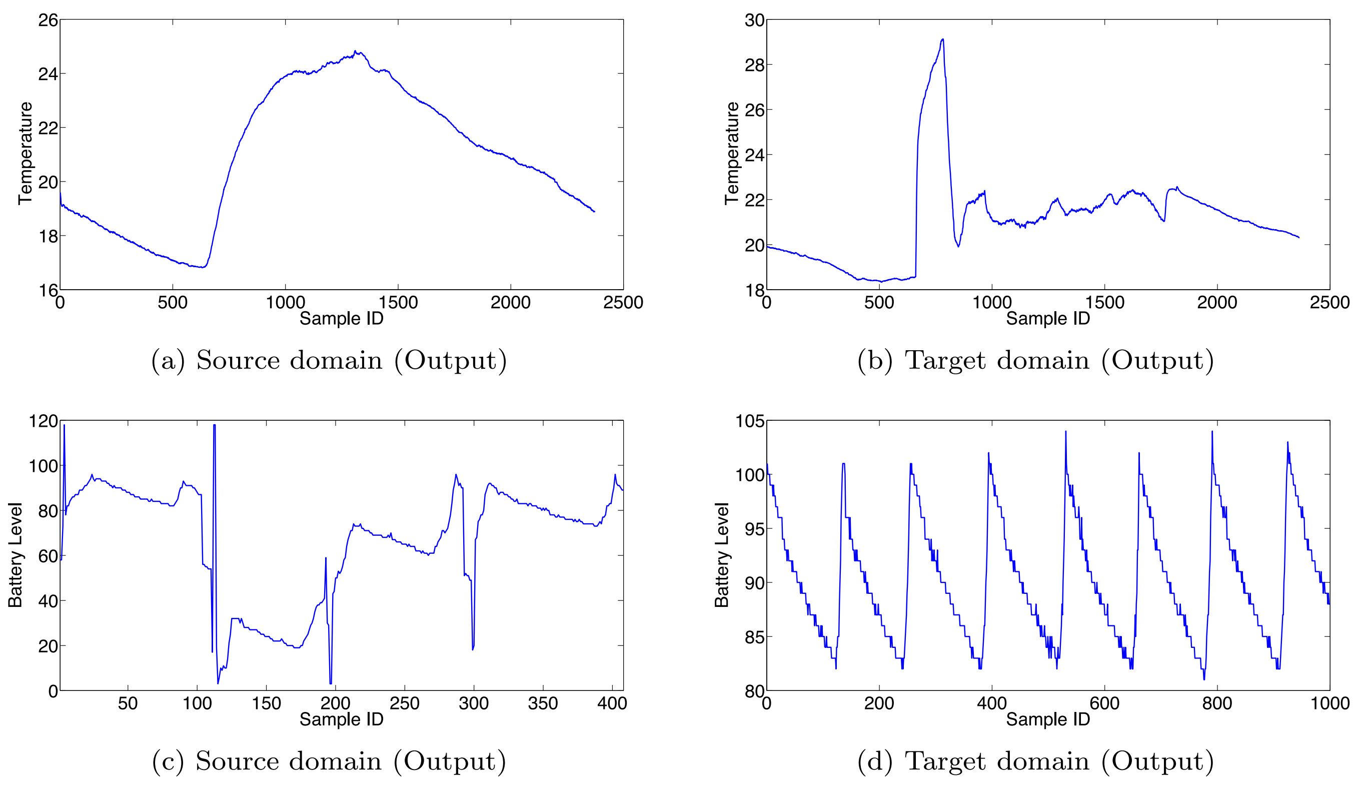


Fig. 2. Intel Laboratory data (top) and Townsville wireless sensor data (bottom).

where m represents the number of neurons in the hidden layer of autoencoder, $\hat{\rho}_i$ represents the average activation value of a neuron and ρ represents its desired value. Ω_ρ is also known as the Kullback–Leibler divergence of $\hat{\rho}_i$ and ρ. The autoencoder is trained using scaled conjugate gradient descent algorithm (Møller, 1993).

The first autoencoder gets the source domain features as inputs and produces an abstraction. In the second autoencoder, this abstracted information is fed as input and a second abstraction is produced. We have found 2 autoencoders worked well for our purpose. However, more or less layers of autoencoder might be required depending on the dataset. This is an optimization step in our framework and needs further research to find out the optimal number of autoencoders in the pre-processing step. This is essentially part of a more fundamental research that attempts to define a good abstraction/representation.

At this point, knowledge transfer mechanism, as described in Section 3 and Algorithm 1, is applied on the derived deep features to transfer the abstract knowledge from source domain to target domain. Finally, conventional adaptive network based fuzzy inference system (ANFIS), interval type-2 fuzzy logic system (IT2FLS) and artificial neural network (ANN) are trained with these features. Once trained, these models are used to predict target domain output. The complete process is shown in Fig. 3. Parameter values used in the process can be found in Table 2. The following aspects of experimental framework are worth noting:

- target domain samples are never processed through autoencoder. Only the number of target domain features (N_T) is passed to deep learning stage to ensure that the number of deep features is equal to the number of features in the target domain.
- target domain samples are never transformed. This is to ensure that we preserve as much information as possible. All of the available target domain samples are used as 'seeds' and therefore, they are only used in the knowledge transfer block.

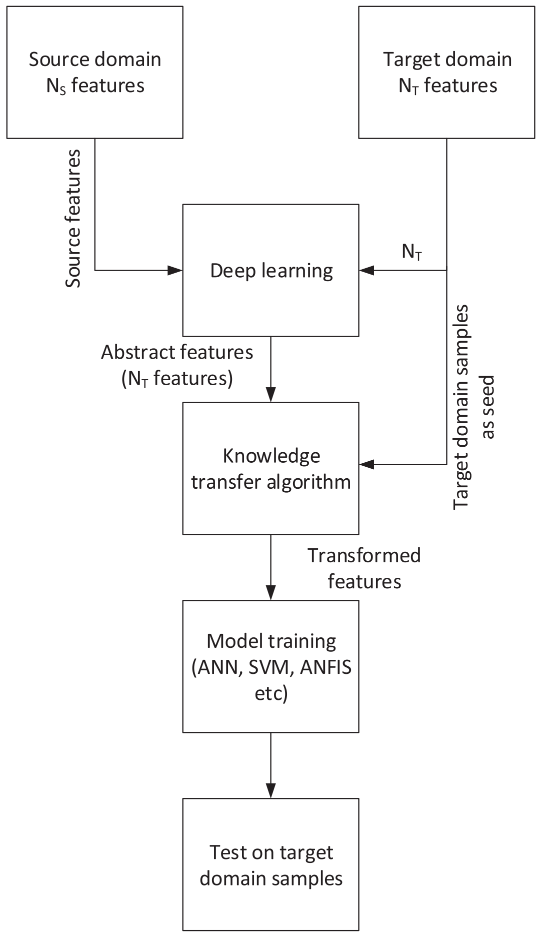

Source domain N_S features
Target domain N_T features
Source features
N_T
Deep learning
Abstract features (N_T features)
Target domain samples as seed
Knowledge transfer algorithm
Transformed features
Model training (ANN, SVM, ANFIS etc)
Test on target domain samples

Fig. 3. Experimental set-up.

Table 2
Parameter values used in experiments.

	Synthetic	Intel	Townsville
Autoencoder (Layer 01)			
Output feature count	6	6	14
Maximum epochs	40,000	40,000	40,000
L2 Weight Regularization	0.004	0.004	0.004
Sparsity Regularization	4	4	4
Sparsity Proportion	0.15	0.15	0.15
Encoder transfer function		Logistic Sigmoidal	
Decoder transfer function		Linear	
Autoencoder (Layer 02)			
Output feature count	2	2	7
Maximum epochs	5000	5000	5000
L2 Weight Regularization	0.002	0.002	0.002
Sparsity Regularization	4	4	4
Sparsity Proportion	0.1	0.1	0.1
Encoder transfer function		Logistic Sigmoidal	
Decoder transfer function		Linear	
ANN			
Hidden layer size	100	100	100
ANFIS			
Number of MF	5	5	5
Maximum epoch	100	100	100

[1] Target domain availability (%): 2, 5, 10, 15, 20, 25. [2] Genetic algorithm (to tune Interval type-2 FLS) parameters: population size: 320, generations: 300. [3] Reported hyper-parameters are finalized after manual trial and error. Only the parameters producing best results are reported.

As mentioned and justified in Section 2, source only baseline is considered as benchmark in this work. In addition to that, a dataset with only deep features is also considered with an attempt to prove that the success of proposed method is not resulting from the usage of deep learning alone. Therefore, 3 different versions of each of the synthetic, Intel lab and Townsville wireless sensor source domain datasets are used to train ANFIS, IT2FLS and ANN models. These 3 versions are denoted as "TL" (adapted/transformed dataset), "SO" (source only dataset) and "DO" (deep learned feature only dataset). For each of these datasets and prediction models, a predetermined percentage of target domain data (i.e. availability of target domain) is used to transform source domain knowledge. In this work, the following availability percentage of target domain data are considered: {2, 5, 10, 15, 20, 25}. For each combination of availability, dataset and prediction models, the experiment is run 5 times to ensure the results are statistically significant and not just a random chance. In each run, the random number generator is shuffled to ensure different target domain data and learning outcome. The test set is always the target domain data, which is never changed throughout the experiment.

6. Result

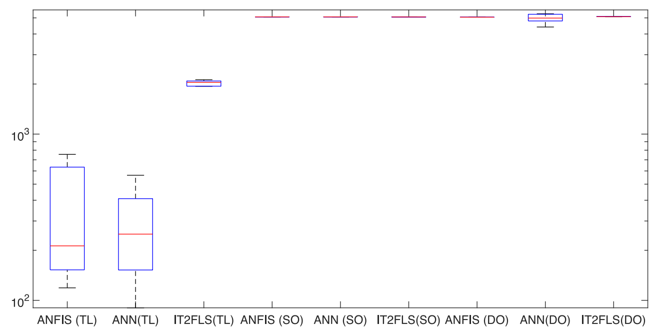
Results of the experiment are presented in Figs. 4–6 and Table 3. The figures represent root mean squared error on the vertical axis. Horizontal axis is divided into three main subgroup. On the very left, the first 3 entries are for learner trained with adapted dataset (proposed method), the middle 3 entries are for the same learners trained with only source domain knowledge and the rightmost 3 entries are for the learners trained with only deep features. Note from Figs. 4–6 that at least one of the first 3 entries is always achieving the lowest error, except for Fig. 5a and b. This clearly shows the superiority of the proposed method. Detailed analysis on each of the datasets is provided below:

6.1. Synthetic data 01

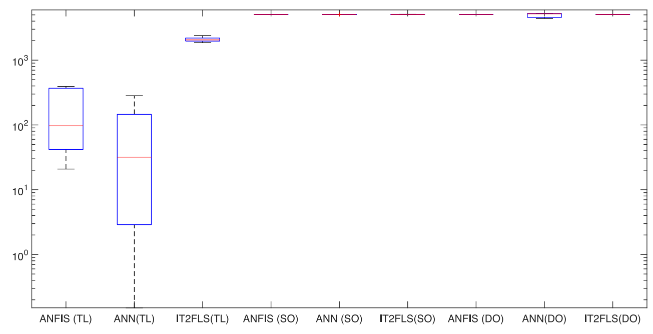
Fig. 4 shows the performance of the proposed adaptation method on the synthetic dataset. The purpose of using such a

Table 3
Median RMSE values for all datasets, generated with 5 runs for each target domain availability (for all models, TL: trained with dataset adapted using our method, SO: trained with source domain dataset only, DO: trained with deep learned features only which shows the performance improvement is not due to deep learning alone % Improvement is calculated with respect to best performing SO or DO model against best performing TL model.).

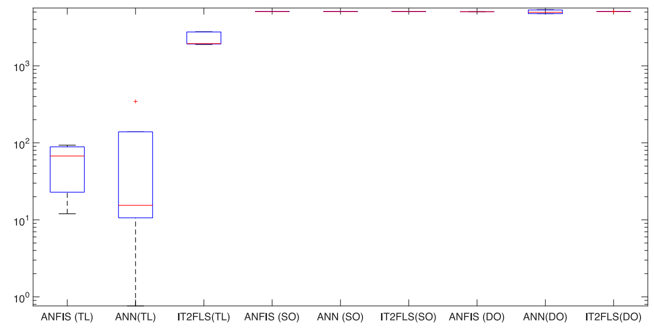
Dataset	Availability	ANFIS			ANN			IT2FLS			Improvement	Improvement	Best TL	Best Other
		TL	SO	DO	TL	SO	DO	TL	SO	DO				
Synthetic	2%	**212.62**	5068.63	5046.1	250.22	5068.49	4983.94	2086.04	5051.13	5074.75	15.03%	15.03	212.62	250.22
	5%	96.75	5068.64	5046.1	**31.88**	5068.49	5259.19	2055.29	5065.67	5044.08	99.37%	99.37	31.88	5044.08
	10%	67.51	5068.64	5046.1	**15.43**	5068.5	4962.06	1947.01	5087.11	5065.05	99.69%	99.69	15.43	4962.06
	15%	**11.49**	5068.64	5046.1	186.45	5068.49	5134.53	2067.41	5086.09	5049.82	99.77%	99.77	11.49	5049.82
	20%	**39.11**	5068.64	5046.1	114.18	5068.49	5015.246	2088.94	5066.88	5056.62	99.22%	99.22	39.11	5046.1
	25%	**10.73**	5068.64	5046.1	47.66	5068.5	5197.73	1872.25	5060.35	5069.08	99.79%	99.79	10.73	5046.1
Intel lab	2%	1.0391	0.4556	**0.18074**	1.8084	0.59827	706.8876	5.6552	1.2461	2.1261	−474.91%	−474.91	1.0391	0.18074
	5%	0.4759	0.4556	**0.1807**	3.5314	0.7551	778.4356	1.4351	1.0222	1.0467	−163.36%	−163.36	0.4759	0.1807
	10%	**0.1683**	0.4555	0.1807	0.8361	1.5328	1247.4	10.8704	0.9403	1.1837	6.86%	6.86	0.1683	0.1807
	15%	**0.0462**	0.4556	0.1807	0.4762	2.7531	1020.05	11.6342	1.0395	2.0098	74.43%	74.43	0.0462	0.1807
	20%	**0.0526**	0.4557	0.1807	0.1985	1.5916	1527.5	3.84	1.3751	1.9852	70.89%	70.89	0.0526	0.1807
	25%	**0.0886**	0.4555	0.1807	0.1299	2.1521	204.5347	8.9646	1.4052	0.9757	50.97%	50.97	0.0886	0.1807
Townsville wireless sensor	2%	2243.32	15.11	25.75	**1.21**	56.82	57.19	2.718	10.41	8.4524	85.68%	85.68	1.21	8.4524
	5%	646.06	15.11	25.75	**1.73**	62.85	63.52	3.48	8.55	5.9	80.93%	80.93	1.73	9.07
	10%	720.45	15.11	25.75	**1.97**	62.09	140.21	3.23	16.77	9.07	86.96%	86.96	1.97	15.11
	15%	425	15.11	25.75	**2.26**	59.01	55.94	6.86	16.86	18.96	58.76%	58.76	2.26	5.48
	20%	74.99	15.11	25.75	**0.196**	94.67	44.13	3.46	5.48	7.86	96.42%	96.42	0.196	5.48
	25%	869.79	15.11	25.73	**0.28**	47.24	76.81	3.37	9.95	12	97.19%	97.19	0.28	9.95



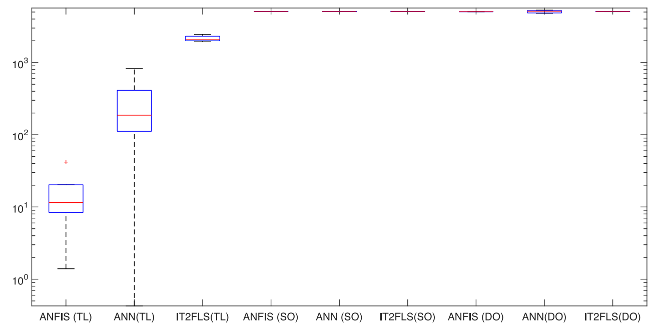
(a) 2% availability



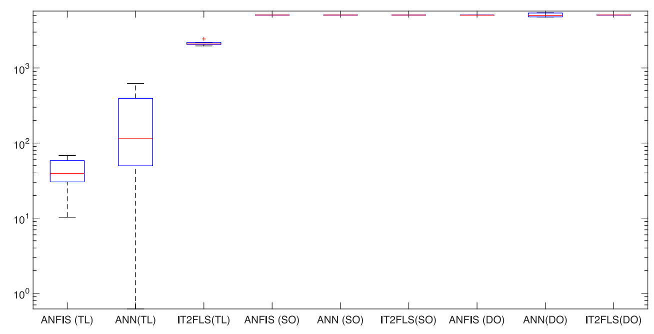
(b) 5% availability



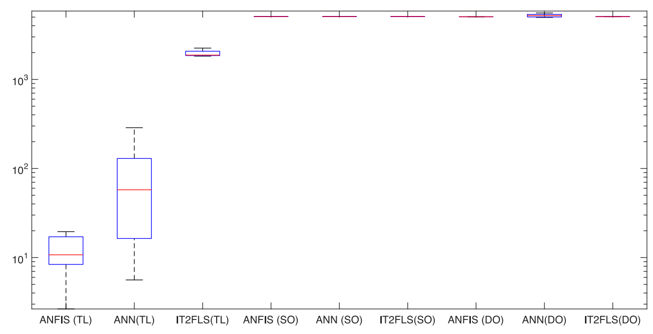
(c) 10% availability



(d) 15% availability



(e) 20% availability



(f) 25% availability

Fig. 4. Performance (RMSE) of adapted model, source data only model and target data only model on the target domain in synthetic dataset. Both the source and target domains have two features. (TL: adapted model, SO: source only model, DO: model trained with deep learned features). Range of error is generated based on 5 iterations. % in the sub-caption denotes availability of target domain data.

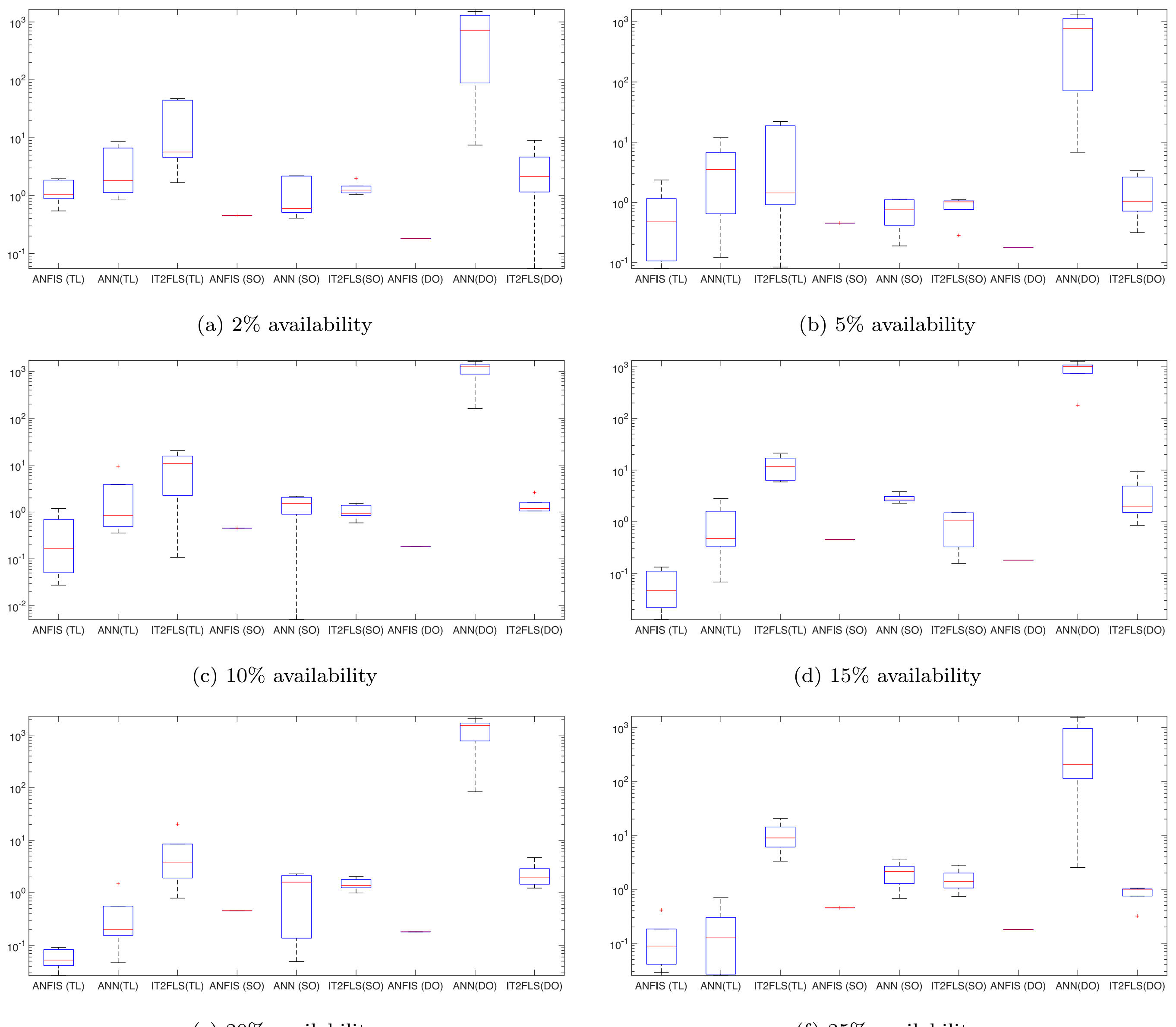
dataset is to examine the potential of our method in controlled environment. Later, the proposed method is tested on real data. Note from Fig. 4 and Table 3 that:

- regardless of the training dataset and prediction scheme, none of the ANFIS, ANN or IT2FLS models is able to provide a good prediction. This is happening because the source domain samples are generated randomly and number of source domain samples (i.e. training samples) are very small. The scarcity of the source domain observations causes difficulty to find an appropriate mapping between inputs and outputs in this random system for all of the models. Finally, number of testing samples in target domain is greater than number of training data samples in source domain which causes the RMSE to be large. This setting is chosen intentionally so that the potential of the proposed adaptation method can be tested in a difficult environment.
- all 3 prediction models trained on source domain data or only deep features provide output with little variance, as evident by the zero or very small range in error boxplot.
- it is clearly seen that the proposed method obtained the best performance compared with its competing methods as it yields the lowest testing error.
- with the increase of target domain availability, the error monotonically decreases for the proposed method whereas in other methods the reduction of RMSE is not large.

6.2. Intel laboratory data

Note from Fig. 5 that:

- Adapted ANFIS model always outperforms the model developed only with source domain data samples i.e. source only model except the first cases where only 2% and 5% target domain data is available. We believe the reason for this anomaly is the very limited target domain data in these two cases so that they are



(a) 2% availability

(b) 5% availability

(c) 10% availability

(d) 15% availability

(e) 20% availability

(f) 25% availability

Fig. 5. Performance (RMSE) of adapted model, source data only model and target data only model on the target domain in Intel Lab dataset. Both the source and target domains have two features. (TL: adapted model, SO: source only model, DO: model trained with deep learned features). Range of error generated based on 5 iterations. % in the sub-caption denotes availability of target domain data.

not fully representative of complete target domain, which contradicts one of the assumptions of this work. Therefore, the extremely small amount of target domain data is misguiding the proposed method.

- Also note that there is considerable difference between source only model and the adapted model (y-axis in the plot is in logarithmic scale). In addition, ANFIS model on adapted data always outperforms the ANN model built with same data when median error value is considered.
- Models built with interval type-2 fuzzy logic system (IT2FLS) do not perform well with the adapted data. However, they consistently performs better than ANN, considering both the median error value and variance of error, in source only and deep learning scenario.
- Performance of ANN built on adapted data gets better in a consistent manner with the increase of target domain data availability.
- Similar observation applies to model built on adapted data using ANFIS as well.
- Most importantly, at least one of the models built with adapted data always outperforms all other models built with source only data and deep learned data, except for the first case. This clearly proves that our proposed method performs well with the Intel laboratory dataset.

Median values of the RMSE for the Intel laboratory data are shown in Table 3. These median values, along with the boxplots, are generated from 5 runs of each target domain data availability. Note from this table that:

- With the adapted dataset, both ANFIS and ANN demonstrate better learning capability and the median error decreases monotonically. However, this is not true for IT2FLS on this dataset. We believe this is due to premature termination of genetic algorithm on the adapted dataset. To ensure fare comparison, GA was run with same configuration (population size: 320,

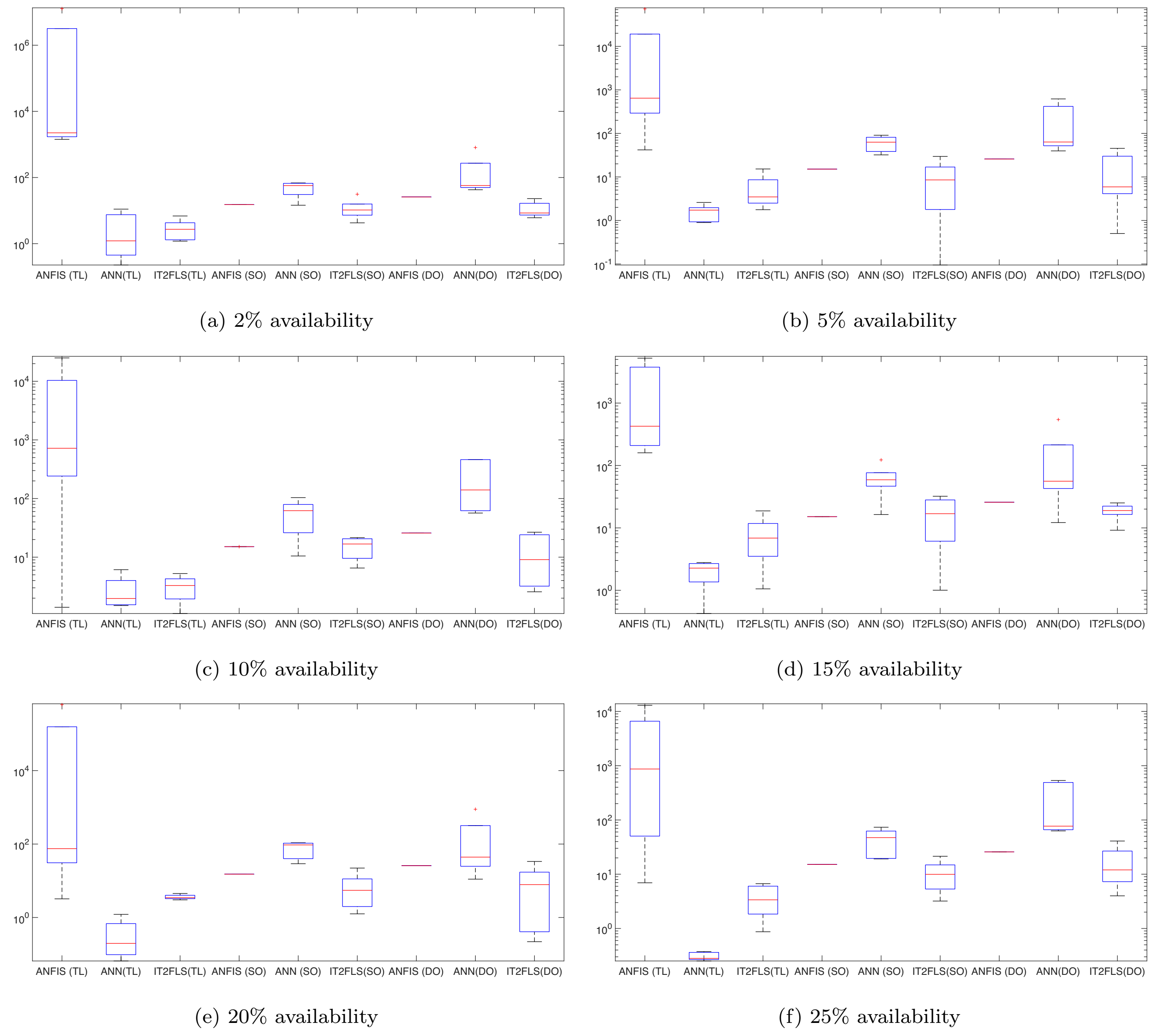


Fig. 6. Performance (RMSE) of adapted model, source data only model and target data only model on the target domain in wireless sensor network, Townsville, Australia. Both the source and target domains have seven features. (TL: adapted model, SO: source only model, DO: model trained with deep learned features). % in the sub-caption denotes availability of target domain data.

generations: 300) for all datasets in order to tune the center of membership functions and standard deviations of UMF and LMF.

- Lowest RMSE values for each of the 6 target domain availability criteria are highlighted in bold. Note that, ANFIS built with adapted dataset is the best performer except first two rows in this table where ANFIS built with only deep features is the winner.
- In addition to that, IT2FLS outperforms ANN on source only and deep only samples except when target domain availability is 2% and 5%, respectively.
- For 2% and 5% target domain data availability, the prediction errors of the proposed method are higher than source only models for ANFIS, ANN and IT2FLS. In addition to that, deep features only model outperforms the proposed method for ANFIS predictions. Naturally, one might ask why the deep features only model outperforms the proposed TL method. The reason for this behavior is not unexpected because:
  - our method assumes that available sample data is fully representative of complete target domain. When the available dataset is too small, the transformation is unable to capture all possible area in target domain and therefore, prediction error becomes high.
  - in addition, the proposed TL method further transforms the deep features based on available target domain samples. Therefore, an insufficient transformation may distort the higher level knowledge gained from deep learned features. This may lead to a higher prediction error than deep features only model.
- For the ANN and ANFIS, increasing target domain data availability implies improved performance. However, IT2FLS performance does not improve in the same fashion.

- ANFIS is roughly producing constant output when trained with only source data or deep learned features. This can be realized by looking at the constant median error values for all availability. Possible reasons for this behavior are unseen range of values in the testing set and no matching rules for the particular combination (as ANFIS does not utilize fully connected rule-base).
- For the same source only dataset or deep learned feature set, ANN is not producing near constant prediction. However, performance of ANFIS is better than ANN on the target domain testing set.
- Also note that, ANN performs significantly better when trained with source domain training data in comparison with the performance of deep feature only training. However, ANFIS performs better when trained with deep learned feature only (in comparison with source knowledge only training).
- ANN trained with deep features only data produces large error compared to the cases when the same ANN is trained with source only knowledge or adapted knowledge. We believe this happens because the ANN structure is not optimized for the deep learned dataset. However, to ensure fare comparison, no alteration is done on ANN for deep knowledge only cases.

6.3. Townsville wireless sensor network

Performance of proposed method on wireless sensor network data in Townsville, Australia is shown in Fig. 6. Since there are 7 inputs (i.e. total number of parameter is 110 which includes 70 nonlinear and 40 linear parameters, number of MFs on each input is 5) in this system with only 365 training data tuples, ANFIS is not able to train properly. However, ANN has no such restriction and is performing very well. In addition, the IT2FLS models, built with Wang-Mendel rule generation method and parameters (i.e. MF centers and standard deviations) tuned with genetic algorithm, is demonstrating good performance on this dataset. Note from Fig. 6 and Table 3 that:

- at least one of the models built with the adapted dataset always outperforms all other models built with source only dataset and deep learned dataset. This clearly demonstrates that our proposed method is not limited to low dimensional dataset.
- ANN trained with adapted dataset is always the best performer among all models and training data. When that same ANN is trained only with source knowledge or deep learned features, test set performance is deteriorated drastically. This clearly proves that the adaptation procedure is contributing to the improved performance.
- IT2FLS trained with adapted dataset always outperforms other IT2FLS models trained with only source knowledge or deep learned features. Again, it proves that adaptation process improves the performance.
- there are large gaps among the median values of ANN trained on adapted data, unmodified source data and deep learned features.
- Similar to the Intel laboratory dataset, ANFIS often produces constant output with source only dataset and deep learned dataset. In addition, ANFIS performs worse than IT2FLS and shows higher standard deviation when trained with adapted dataset.

Finally, as a summary of the above discussions, we want to highlight the following:

- the proposed method is a framework for knowledge transfer. It provides a feature set for any base learner to learn from. Optimization of the base learner is dependent on specific dataset and one particular learner may or may not perform the best on all the adapted feature sets. As long as at least one of the compared learners, when trained with proposed adaptation framework, performs the best, we can reasonably conclude that the proposed methodology outperforms the benchmarks. In practice, not all established learning methods works the same way on all dataset and not one single method exists that outperforms everything else on every dataset.
- our experiment clearly shows that at least one of the base learner trained with adapted dataset outperforms the benchmarks. Even though no single base learner (e.g. ANN) outperforms everything else in every dataset, we can still reasonably conclude the proposed framework outperforms the benchmarks as our proposed method is independent of the base learner.

6.4. Significance testing

We have provided p-values of the Wilcoxon rank sum test in Table 4. This table is generated by considering the best performing learner (i.e. ANN, ANFIS or IT2FLS) for a given dataset and target domain availability. Afterwards, the performance of the best learner is compared with the same learner trained with 2 remaining datasets. For example, in the synthetic dataset with 10% availability, ANN trained with adapted dataset is producing the lowest RMSE error. Therefore, performance of this best performing ANN is compared with that of two other ANNs trained with 'SO' and 'DO' benchmark data. Since our proposed method does not deal with a learner, but the adaptation procedure, it is only logical to compare the same learner across different training sets to understand the significance. Note from Table 4 that proposed adaptation procedure is outperforming the benchmarks in a statistically significant way for 11 out of the 18 cases. In 3 cases, the performance improvement is not statistically significant under the probability cutoff of 0.05. Deep feature only benchmark outperforms the proposed method in 4 cases, one of which is not a statistically significant performance.

6.5. Limitations and future works

Although our proposed method have shown good performance on both synthetic and real dataset for regression, it does come with some limitations. One of the key assumption in the process is the available, scarce dataset will be representative of the target domain. This implies that we need at least one sample from all possible cases in target domain (in classification set-up, this would dictate to have at least one sample per class). If this assumption is not true, performance of the proposed method will not improve at all. The extent of such performance will depend on extent of missing scenarios in available target domain samples.

Another assumption of the proposed method is the deep learned feature will provide a better and abstract representation of source domain. Although this assumption is generally true, it may not hold if the source domain don't have enough samples or the deep learning method is not optimized well. This work utilizes a simple form of deep learning, namely auto-encoder. However, we did not test the performance of other deep learning methods e.g. convolutional neural network (CNN) or recurrent neural network (RNN).

Another limitation of the proposed method is that it does not work with classification problems. This stemmed from the fact that no method to transfer the output/target from source domain is proposed. Essentially, the algorithm transform only the input side of source domain dataset.

Furthermore, the choice of clustering algorithm and deep learning method (along with corresponding parameters) may influence the outcome of the proposed algorithm. Impact of clustering and

Table 4
Statistical significance of performance comparisons. Bold rows indicate that a learning method trained with proposed adaptation method outperforms other models in a statistically significant manner.

Dataset	Availability	*P*-value	Null Hypothesis (H_0)	Best performer	Significant	Effect size
Synthetic	2%	**0.0080**	$ANFIS_{TL} = ANFIS_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANFIS_{TL} = ANFIS_{DO}$	**TL**	**TRUE**	0.7618
	5%	**0.0080**	$ANN_{TL} = ANN_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANN_{TL} = ANN_{DO}$	**TL**	**TRUE**	0.7618
	10%	**0.0080**	$ANN_{TL} = ANN_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANN_{TL} = ANN_{DO}$	**TL**	**TRUE**	0.7618
	15%	**0.0080**	$ANFIS_{TL} = ANFIS_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANFIS_{TL} = ANFIS_{DO}$	**TL**	**TRUE**	0.7618
	20%	**0.0080**	$ANFIS_{TL} = ANFIS_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANFIS_{TL} = ANFIS_{DO}$	**TL**	**TRUE**	0.7618
	25%	**0.0080**	$ANFIS_{TL} = ANFIS_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANFIS_{TL} = ANFIS_{DO}$	**TL**	**TRUE**	0.7618
Intel Lab	2%	0.0080	$ANFIS_{TL} = ANFIS_{DO}$	DO	TRUE	0.7618
		0.0080	$ANFIS_{SO} = ANFIS_{DO}$	DO	TRUE	0.7618
	5%	0.6825	$ANFIS_{TL} = ANFIS_{DO}$	DO	FALSE	0.1501
		0.0080	$ANFIS_{SO} = ANFIS_{DO}$	DO	TRUE	0.7618
	10%	0.6905	$ANFIS_{TL} = ANFIS_{SO}$	TL	FALSE	0.1573
		0.6825	$ANFIS_{TL} = ANFIS_{DO}$	TL	FALSE	0.1501
	15%	**0.0080**	$ANFIS_{TL} = ANFIS_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANFIS_{TL} = ANFIS_{DO}$	**TL**	**TRUE**	0.7618
	20%	**0.0080**	$ANFIS_{TL} = ANFIS_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANFIS_{TL} = ANFIS_{DO}$	**TL**	**TRUE**	0.7618
	25%	**0.0080**	$ANFIS_{TL} = ANFIS_{SO}$	**TL**	**TRUE**	0.7618
		0.1270	$ANFIS_{TL} = ANFIS_{DO}$	TL	FALSE	0.3607
Townsville wireless sensor	2%	**0.0080**	$ANN_{TL} = ANN_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANN_{TL} = ANN_{DO}$	**TL**	**TRUE**	0.7618
	5%	**0.0080**	$ANN_{TL} = ANN_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANN_{TL} = ANN_{DO}$	**TL**	**TRUE**	0.7618
	10%	**0.0080**	$ANN_{TL} = ANN_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANN_{TL} = ANN_{DO}$	**TL**	**TRUE**	0.7618
	15%	**0.0080**	$ANN_{TL} = ANN_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANN_{TL} = ANN_{DO}$	**TL**	**TRUE**	0.7618
	20%	**0.0080**	$ANN_{TL} = ANN_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANN_{TL} = ANN_{DO}$	**TL**	**TRUE**	0.7618
	25%	**0.0080**	$ANN_{TL} = ANN_{SO}$	**TL**	**TRUE**	0.7618
		0.0080	$ANN_{TL} = ANN_{DO}$	**TL**	**TRUE**	0.7618

[1] Many p-values are equal because the error outcomes of many different approaches are constant. For example in the first two cases in the INTEL lab dataset, in testing $ANFIS_{TL}$ vs $ANFIS_{SO}$, the $ANFIS_{SO}$ approach produces a set of 5 outcomes that are all at 0.4555. In another test, i.e. $ANFIS_{TL}$ vs $ANFIS_{DO}$, the $ANFIS_{DO}$ generates a set of 5 outcomes that are all at 0.1807. Because of this, even though $ANFIS_{SO}$ and $ANFIS_{DO}$ have different RMSE values, the p-values in the two aforementioned significant tests are the same at 0.0080. [2] Only the best performing learning method is picked for comparison against the same model trained with different dataset. For example, if ANN is the best performer for a given dataset and target domain availability, then the performance of ANN is considered in statistical test. In this case, ANN is trained and compared with adapted training set (TL), only source knowledge (SO) and deep learned features (DO). [3] Learning method and training dataset are denoted using subscript notation. For example, ANN_{TL} indicates an artificial neural network trained with adapted dataset using proposed method. [4] Since employed Wilcoxon rank sum test does not care about the ordering of input variables, corresponding RMSE values must be considered while deciding the best performer. [5] $ANN_x = ANN_y$ in the null hypothesis implies that performance of an ANN trained with dataset x has the same median error as another ANN trained with dataset y [6] Effect size $>= 0.5$ is considered medium whereas values greater than 0.8 is considered large effect size (Cohen (1988))

deep learning method will be investigated as part of upcoming future works. In addition, we would also investigate the impact of difference between the abstracted feature space and original target domain feature space.

7. Conclusion

In this work, a new method is proposed to transform the source dataset to improve prediction performance in target domain. A limited amount of target domain data is used to accomplish this transformation. The proposed method is independent of regression methods used for prediction. Therefore, it presents a new opportunity to improve the performance of existing transfer learning methods. Effectiveness of the proposed method is demonstrated using one synthetic and two real datasets.

Acknowledgement

This work is supported by the Institute of Intelligent Systems Research and Innovation at Deakin University.

References

Afrin, K., Nepal, B., & Monplaisir, L. (2018). A data-driven framework to new product demand prediction: Integrating product differentiation and transfer learning approach. *Expert Systems with Applications*. doi:10.1016/j.eswa.2018.04.032.
Aghamaleki, J. A., & Baharlou, S. M. (2018). Transfer learning approach for classification and noise reduction on noisy web data. *Expert Systems with Applications, 105*, 221–232. doi:10.1016/j.eswa.2018.03.042.
Al-Mubaid, H., Umair, S., et al. (2006). A new text categorization technique using distributional clustering and learning logic. *Knowledge and Data Engineering, IEEE Transactions on, 18*(9), 1156–1165.
Ando, R. K., & Zhang, T. (2005). A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd annual meeting on association for computational linguistics* (pp. 1–9). Association for Computational Linguistics.
Argyriou, A., Evgeniou, T., & Pontil, M. (2008). Convex multi-task feature learning. *Machine Learning, 73*(3), 243–272.
Argyriou, A., Pontil, M., Ying, Y., & Micchelli, C. A. (2007). A spectral regularization framework for multi-task structure learning. In *Advances in neural information processing systems* (pp. 25–32).
Bengio, Y. (2012). Deep learning of representations for unsupervised and transfer learning. *Unsupervised and Transfer Learning Challenges in Machine Learning, 7*, 19.
Bickel, S., Brückner, M., & Scheffer, T. (2007). Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on machine learning* (pp. 81–88). ACM.
Blitzer, J., Dredze, M., Pereira, F., et al. (2007). Biographies, bollywood, boom-boxes

and blenders: Domain adaptation for sentiment classification. In *ACL: 7* (pp. 440–447).
Blitzer, J., McDonald, R., & Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing* (pp. 120–128). Association for Computational Linguistics.
Bonilla, E. V., Chai, K. M., & Williams, C. (2007). Multi-task gaussian process prediction. In *Advances in neural information processing systems* (pp. 153–160).
Calais Guerra, P. H., Veloso, A., Meira Jr, W., & Almeida, V. (2011). From bias to opinion: a transfer-learning approach to real-time sentiment analysis. In *Proceedings of the 17th acm sigkdd international conference on knowledge discovery and data mining* (pp. 150–158). ACM.
Cohen, J. (1988). *Statistical power analysis for the behavioral sciences.* (2nd).
Dai, W., Xue, G.-R., Yang, Q., & Yu, Y. (2007a). Co-clustering based classification for out-of-domain documents. In *Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 210–219). ACM.
Dai, W., Xue, G.-R., Yang, Q., & Yu, Y. (2007b). Transferring naive Bayes classifiers for text classification. In *Proceedings of the national conference on artificial intelligence: 22* (p. 540). Menlo Park, CA; Cambridge, MA; London: AAAI Press; MIT Press 1999;.
Dai, W., Yang, Q., Xue, G.-R., & Yu, Y. (2007c). Boosting for transfer learning. In *Proceedings of the 24th international conference on machine learning* (pp. 193–200). ACM.
Daumé III, H. (2009). Frustratingly easy domain adaptation. arXiv:0907.1815.
Davis, J., & Domingos, P. (2009). Deep transfer via second-order Markov logic. In *Proceedings of the 26th annual international conference on machine learning* (pp. 217–224). ACM.
Demuth, H. B., Beale, M. H., De Jess, O., & Hagan, M. T. (2014). *Neural network design*. Martin Hagan.
Evgeniou, A., & Pontil, M. (2007). Multi-task feature learning. *Advances in Neural Information Processing Systems, 19*, 41.
Evgeniou, T., & Pontil, M. (2004). Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 109–117). ACM.
Falkenhainer, B., Forbus, K. D., & Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial Intelligence, 41*(1), 1–63.
Fan, W., Davidson, I., Zadrozny, B., & Yu, P. S. (2005). An improved categorization of classifier's sensitivity on sample selection bias. In *Data mining, fifth IEEE international conference on* (p. 4). IEEE.
Fung, G. P. C., Yu, J. X., Lu, H., & Yu, P. S. (2006). Text classification without negative examples revisit. *Knowledge and Data Engineering, IEEE Transactions on, 18*(1), 6–20.
Gao, J., Fan, W., Jiang, J., & Han, J. (2008). Knowledge transfer via multiple model local structure mapping. In *Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 283–291). ACM.
Gharehbaghi, A., & Lindén, M. (2017). A deep machine learning method for classifying cyclic time series of biological signals using time-growing neural network. *IEEE Transactions on Neural Networks and Learning Systems*.
Ghiassi, M., & Lee, S. (2018). A domain transferable lexicon set for twitter sentiment analysis using a supervised machine learning approach. *Expert Systems with Applications, 106*, 197–216. doi:10.1016/j.eswa.2018.04.006.
Glorot, X., Bordes, A., & Bengio, Y. (2011). Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)* (pp. 513–520).
Han, D., Liu, Q., & Fan, W. (2018). A new image classification method using cnn transfer learning and web data augmentation. *Expert Systems with Applications, 95*, 43–56. doi:10.1016/j.eswa.2017.11.028.
Haykin, S., & Network, N. (2004). A comprehensive foundation. *Neural Networks, 2*(2004), 41.
Hossain, I., Khosravi, A., Hettiarachchi, I., & Nahavandi, S. (2018). Multiclass informative instance transfer learning framework for motor imagery-based brain-computer interface. *Computational Intelligence and Neuroscience, 2018*.
Hossain, I., Khosravi, A., & Nahavandi, S. (2016). Active transfer learning and selective instance transfer with active learning for motor imagery based bci. In *Neural networks (IJCNN), 2016 international joint conference on* (pp. 4048–4055). IEEE.
Huang, J., Gretton, A., Borgwardt, K. M., Schölkopf, B., & Smola, A. J. (2006). Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems* (pp. 601–608).
Jang, J.-S. (1993). Anfis: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics, 23*(3), 665–685.
Jebara, T. (2004). Multi-task feature and kernel selection for svms. In *Proceedings of the twenty-first international conference on machine learning* (p. 55). ACM.
Jiang, J., & Zhai, C. (2007). Instance weighting for domain adaptation in nlp. In *ACL: 7* (pp. 264–271).
Khatami, A., Babaie, M., Tizhoosh, H., Khosravi, A., Nguyen, T., & Nahavandi, S. (2018). A sequential search-space shrinking using CNN transfer learning and a radon projection pool for medical image retrieval. *Expert Systems with Applications, 100*, 224–233. doi:10.1016/j.eswa.2018.01.056.
Lawrence, N. D., & Platt, J. C. (2004). Learning to learn with the informative vector machine. In *Proceedings of the twenty-first international conference on machine learning* (p. 65). ACM.
Lee, S.-I., Chatalbashev, V., Vickrey, D., & Koller, D. (2007). Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 24th international conference on machine learning* (pp. 489–496). ACM.
Liang, Q., & Mendel, J. M. (1999). An introduction to type-2 tsk fuzzy logic systems. In *Fuzzy systems conference proceedings, 1999. fuzz-IEEE'99. 1999 ieee international: 3* (pp. 1534–1539). IEEE.
Liang, Q., & Mendel, J. M. (2000). Interval type-2 fuzzy logic systems: Theory and design. *Fuzzy Systems, IEEE Transactions on, 8*(5), 535–550.
Liao, X., Xue, Y., & Carin, L. (2005). Logistic regression with an auxiliary data source. In *Proceedings of the 22nd international conference on machine learning* (pp. 505–512). ACM.
Mendel, J. M. (2007). Type-2 fuzzy sets and systems: An overview. *Computational Intelligence Magazine, IEEE, 2*(1), 20–29.
Mesnil, G., Dauphin, Y., Glorot, X., Rifai, S., Bengio, Y., Goodfellow, I. J., et al. (2012). Unsupervised and transfer learning challenge: A deep learning approach.. *ICML Unsupervised and Transfer Learning, 27*, 97–110.
Mihalkova, L., Huynh, T., & Mooney, R. J. (2007). Mapping and revising Markov logic networks for transfer learning. In *Aaai: 7* (pp. 608–614).
Mihalkova, L., & Mooney, R. J. (2008). Transfer learning by mapping with minimal target data. In *Proceedings of the aaai-08 workshop on transfer learning for complex tasks*.
Mohring, K., Myers, T., Atkinson, I., VanderWal, J., & van der Valk, S. (2015). Sensors in heat: A pilot study for high resolution urban sensing in an integrated streetlight platform. In *Intelligent sensors, sensor networks and information processing (issnip), 2015 IEEE tenth international conference on* (pp. 1–6). IEEE.
Mohring, K. A. (2014). *Walker stalker v2 rooftop deployment test trial wireless sensor network*. James Cook University, Townsville, Australia. doi:10.4225/28/546BCED9674BB.
Møller, M. F. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks, 6*(4), 525–533.
Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on, 22*(10), 1345–1359.
Pan, S. J., Zheng, V. W., Yang, Q., & Hu, D. H. (2008). Transfer learning for wifi-based indoor localization. In *Proc. workshop transfer learning for complex task of the 23rd assoc. for the advancement of artificial intelligence (aaai) conf. artificial intelligence*.
Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the acl-02 conference on empirical methods in natural language processing-volume 10* (pp. 79–86). Association for Computational Linguistics.
Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., & Lawrence, N. D. (2009). *Dataset shift in machine learning*. The MIT Press.
Raina, R., Battle, A., Lee, H., Packer, B., & Ng, A. Y. (2007). Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on machine learning* (pp. 759–766). ACM.
Sarinnapakorn, K., & Kubat, M. (2007). Combining subclassifiers in text categorization: A dst-based solution and a case study. *Knowledge and Data Engineering, IEEE Transactions on, 19*(12), 1638–1651.
Schwaighofer, A., Tresp, V., & Yu, K. (2004). Learning gaussian process kernels via hierarchical bayes. In *Advances in neural information processing systems* (pp. 1209–1216).
Shell, J., & Coupland, S. (2015). Fuzzy transfer learning: Methodology and application. *Information Sciences, 293*, 59–79.
Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P. V., & Kawanabe, M. (2008). Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in neural information processing systems* (pp. 1433–1440).
Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, (1), 116–132.
Tan, S., Wu, G., Tang, H., & Cheng, X. (2007). A novel scheme for domain-transfer problem in the context of sentiment analysis. In *Proceedings of the sixteenth ACM conference on conference on information and knowledge management* (pp. 979–982). ACM.
Wang, C., & Mahadevan, S. (2008). Manifold alignment using procrustes analysis. In *Proceedings of the 25th international conference on machine learning* (pp. 1120–1127). ACM.
Zadeh, L. A. (1965). Fuzzy sets. *Information and Control, 8*(3), 338–353.
Zadrozny, B. (2004). Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on machine learning* (p. 114). ACM.