Energy & Buildings 165 (2018) 352–363

Contents lists available at ScienceDirect

Energy & Buildings

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

Transfer learning with seasonal and trend adjustment for cross-building energy forecasting

Mauro Ribeiro[a], Katarina Grolinger[a,*], Hany F. ElYamany[a,b], Wilson A. Higashino[a], Miriam A.M. Capretz[a]

[a] Department of Electrical and Computer Engineering, Western University, London, ON N6A 5B9, Canada
[b] Department of Computer Science, Suez Canal University, Ismailia, Egypt

ARTICLE INFO

Article history:
Received 29 August 2017
Revised 1 December 2017
Accepted 22 January 2018
Available online 3 February 2018

Keywords:
Energy forecasting
Transfer learning
Cross-building forecasting
Seasonal adjustment
Trend adjustment
Energy consumption

ABSTRACT

Large scale smart meter deployments have resulted in popularization of sensor-based electricity forecasting which relies on historical sensor data to infer future energy consumption. Although those approaches have been very successful, they require significant quantities of historical data, often over extended periods of time, to train machine learning models and achieve accurate predictions. New buildings and buildings with newly installed meters have small historical datasets that are insufficient to create accurate predictions. Transfer learning methods have been proposed as a way to use cross-domain datasets to improve predictions. However, these methods do not consider the effects of seasonality within domains. Consequently, this paper proposes Hephaestus, a novel transfer learning method for cross-building energy forecasting based on time series multi-feature regression with seasonal and trend adjustment. This method enables energy prediction with merged data from similar buildings with different distributions and different seasonal profiles. Thus, it improves energy prediction accuracy for a new building with limited data by using datasets from other similar buildings. Hephaestus works in the pre- and post- processing phases and therefore can be used with any standard machine learning algorithm. The case study presented here demonstrates that the proposed approach can improve energy prediction for a school by 11.2% by using additional data from other schools.

1. Introduction

The world today relies on data to drive discovery, optimize processes, and improve decision making. Sensors are becoming cheap and popular, enabling everyone to collect new data and create new applications. This is leading to an explosive growth of data in terms of the number of attributes, data points and datasets.

In the building sector, smart meters that measure and communicate electricity consumption have become widely deployed. They create opportunities for new energy applications and possibilities to apply sensor-based energy forecasting on a large scale. Sensor-based forecasting relies on historical data from smart meters or sensors to infer future energy consumption. It uses machine learning techniques such as Support Vector Regression [1], Neural Networks [2], and autoregressive integrated moving average (ARIMA) models [2].

In machine learning, the quantity and complexity of data required to provide accurate predictions depend on data randomness and number of features [3,4]. Data randomness represents lack of pattern, while features are attributes that are correlated with the labels. In general, more data translate into more accurate machine learning models. Therefore, to accurately forecast energy consumption for a building using a machine learning approach, a sufficient quantity of historical data from the same building is needed.

The challenge is that building managers are eager for accurate predictions as early as possible, but new buildings have small historical data sets. Therefore, this work explores sensor-based forecasting for a building with limited historical data by using consumption data from other similar buildings. The goal is to increase data variance, to fill in gaps due to missing samples, and consequently to create more accurate predictions.

The solution would be simple if the datasets considered were in the same domain and could be represented by the same models; a standard machine learning algorithm could be used to analyze all datasets as one. However, this is not the case because building consumption data are gathered in diverse contexts. For example,

* Corresponding author.
E-mail address: kgroling@uwo.ca (K. Grolinger).

https://doi.org/10.1016/j.enbuild.2018.01.034
0378-7788/

if one building uses incandescent lights and another uses fluorescent lights, their energy consumptions cannot be represented by a single predictive model without adjustments. In addition, the seasons of the year and human activities may result in distinct seasonal patterns (e.g., one building can have peaks during Mondays, but another on Fridays, depending on the routine that each one follows). Therefore, to analyze different buildings together, their seasonality differences must also be considered. Similarly, trend, a long-term variation (direction) of the data, also can be different among buildings and needs to be accounted for.

Approaches collectively known as transfer learning address the challenge of transferring extracted knowledge from one domain to another [5]. Nevertheless, these approaches do not consider the seasonality effects present in electricity data and usually cannot be used with diverse machine learning techniques.

To overcome these limitations, this paper introduces Hephaestus, a transfer learning method with seasonal and trend adjustment for cross-building energy prediction. Hephaestus can be applied to buildings with small data sets by leveraging data from other similar buildings. Moreover, it works with any standard machine learning algorithm. The proposed solution is evaluated on a case study involving energy prediction for a school by using additional data from other schools. The results obtained show that prediction accuracy is significantly improved compared to using only data from the target school, or to using all data, but without Hephaestus.

This paper is organized as follows: Section 2 reviews related concepts in normalization, time series, and transfer learning. Section 3 reviews related work, and Section 4 describes the proposed Hephaestus method. Section 5 presents the energy consumption case study, and, finally, Section 6 concludes the paper.

2. Background

This section introduces the concepts of normalization, time series analysis, and transfer learning that were used to design Hephaestus.

2.1. Normalization

Normalization is defined in this paper as follows:

Definition 1 (Normalization)**.** Given two different sets of values S_1 and S_2, a normalization Ψ is a linear transformation where $\Psi(S_1)$ and $\Psi(S_2)$ are in the same domain and have similar scales.

It is the process of aligning values measured and stored at different scales or proportions to a common scale so that they can be compared and operated on together. In machine learning, normalization is an essential pre-processing step and can significantly improve model performance.

Min-max is one of the most popular normalization methods in machine learning [6,7]. It is commonly referred to simply as "normalization" or sometimes as "feature scaling" and is represented by the equation:

$$\text{min-max} = \frac{x - X_{\min}}{X_{\max} - X_{\min}} \tag{1}$$

where x is the current value, and $X_{\min}$ and $X_{\max}$ are the minimum and maximum values of the entire dataset. The min-max method rescales values and confine samples to an interval between 0 and 1.

The z-score, also known as the standard score or standardization, is another normalization method. It is represented by the equation:

$$z = \frac{x - \mu}{\sigma} \tag{2}$$

which returns the distance of x from the mean μ, measured in multiples of the standard deviation σ [8]. The use of z-score normalization is recommended for attributes that follow a normal distribution.

Unlike min-max, the z-score does not limit values to lie between a specific minimum and maximum. Instead, it maintains the normal distribution, in which most values are statistically concentrated within a range of z-scores. Because a z-score can be any real number, outliers can still have higher values that are not limited by the minimum and maximum values.

2.2. Time series regression model and seasonal adjustment

A time series regression model is used to predict new values based on time series data, which contain successive measurements at different points in time. These models are used in many areas, including revenue prediction [9] and energy consumption forecasting [10,11].

A time series regression model can be either additive [12]:

$$y_t = T_t + C_t + S_t + I_t \tag{3}$$

or multiplicative

$$y_t = T_t \times C_t \times S_t \times I_t \tag{4}$$

The trend component (T_t) is a smooth, regular, and long-term statistical series and represents a general growth or decline; the cyclical (C_t) component is a pattern that occurs repeatedly several times during an irregular period; the seasonal component (S_t) is similar to the cyclical component, but the pattern occurs in a well-defined period (e.g., daily, weekly, or monthly); and the irregular component (I_t) is the remainder, which can be related to non-temporal factors or simply considered an error. The cyclical (C_t) and trend (T_t) components are often analyzed together and merged as a single trend-cycle component (TC_t) [12,13].

Choosing the most appropriate model depends on the data to be analyzed. The additive model is most suitable when the *variation* (the distance between highs and lows) remains relatively constant over time; the multiplicative model is most useful when the variation changes with the local average (the higher the average, the higher the variation) [13].

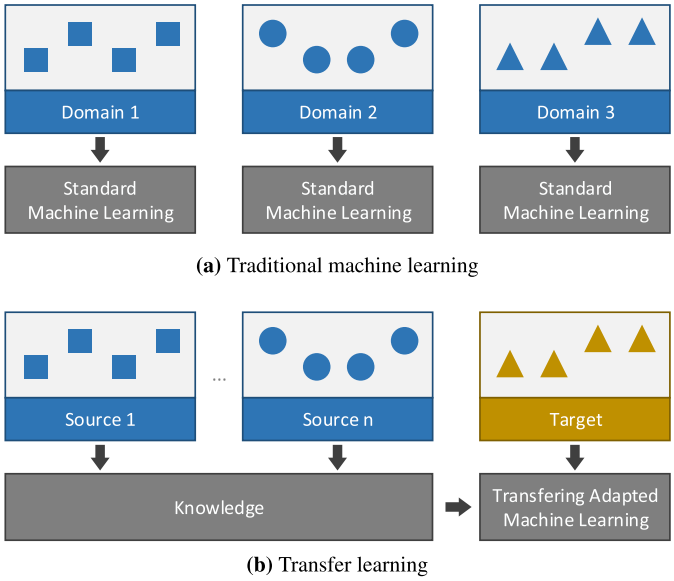
Seasonal adjustment is a procedure to improve the properties of the parameter estimates for time series regression [12]. In this procedure, past observations are used to calculate a trend factor and a seasonal index as estimate of the trend-cycle (TC_t) and seasonal (S_t) components. The trend factor and seasonal index are then used to remove the respective components from the time series, enabling data analysis without these components and later adjustment of the prediction.

2.3. Transfer learning

Transfer learning aims to improve the learning task in a target domain using the knowledge from other domains and learning tasks [5]. It is defined as follows:

Definition 2 (Transfer Learning)**.** Given a source domain $\mathcal{D}_S$ and a learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and a learning task $\mathcal{T}_T$, transfer learning aims to improve the learning of the target predictive function $r_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$ [5,14].

From the definition above, a domain is defined as a pair $\mathcal{D} = \{\mathcal{F}, P(X)\}$, where $\mathcal{F} = \{f_1, ., f_n\}$ is a feature space with n dimensions, f_k is a feature, X is a learning sample such that $X = \{x_1, \ldots, x_n\} \in \mathcal{F}$ and $P(X)$ is the marginal probability distribution of X. For domains to be considered different it is sufficient that ei-



(a) Traditional machine learning

(b) Transfer learning

Fig. 1. Types of machine learning approaches.

ther feature spaces or marginal probability distributions are different. A task is a pair $\mathcal{T} = \{\mathcal{Y}, r(\cdot)\}$, where $\mathcal{Y}$ is the label space and $r(\cdot)$ is the predictive function. From a probabilistic viewpoint, $r(X)$ can also be written as the conditional probability distribution $P(Y|X)$ [5].

Fig. 1 compares different types of machine learning approaches. Fig. 1(a) shows how a standard machine learning algorithm works, using only a single domain for each model to learn from separately. Fig. 1(b) demonstrates how transfer learning differs from standard machine learning, using the knowledge extracted from multiple source domains to improve the prediction for a target domain.

Transfer learning has three main categories [5,15]: (a) inductive transfer learning, where the target task is different from the source task ($\mathcal{T}_S \neq \mathcal{T}_T$); (b) transductive learning, where the tasks are the same ($\mathcal{T}_S = \mathcal{T}_T$), but the domains are different ($\mathcal{D}_S \neq \mathcal{D}_T$). In particular, when the feature spaces of the domains are the same ($\mathcal{F}_S = \mathcal{F}_T$) but the distributions are different ($P(X_S) \neq P(X_T)$), this category is also known as domain adaptation; and (c) unsupervised transfer learning, which is similar to inductive transfer learning for unsupervised learning tasks, where no labeled data are available.

Transfer learning can also be grouped according to what kind of knowledge is transferred across tasks [5]:

(a) Instance-based, where labeled data are selected and reweighted from the source domain to be used in the target domain;
(b) Feature representation-based, where a new feature space is composed to satisfy all the domains;
(c) Parameter-based, where the parameters used to train the source are used to train the target; and
(d) Relational knowledge-based, where a mapping of relational knowledge is built between the source and target domains.

3. Related work

In this paper, related work is divided into two subsections. The first subsection discusses energy forecasting methods, including those used in the building sector. The second subsection presents transfer learning studies within different domains and based on various aspects such as domain generalization and seasonal adjustment.

3.1. Energy forecasting

The importance of energy forecasting together with large-scale deployments of smart meters and other sensors that can capture energy consumption at short intervals have resulted in increased interest in sensor-based forecasting. Although in the past, energy consumption was predicted on annual or monthly bases, today forecasting has moved towards hourly or even shorter intervals [1,16].

Research studies have investigated energy prediction in residential [17], institutional [18], commercial [16], event venues [19], and other buildings. These studies predicted future energy consumption based on past energy consumption data and various contextual attributes such as weather, operating hours, and occupancy. In most sensor-based forecasting research, including the studies just mentioned, energy prediction for a building relies on past energy data from the same building because it is expected that future energy consumption patterns will follow historical ones. Nevertheless, to carry out such forecasting, a large quantity of past data is needed, possibly over an extended time period to capture different weather and seasons. Hence, such solutions cannot be used for new buildings where very few historical data points are available. The work described here investigates if and how energy forecasting can be performed for a new building with very limited historical data, but using data available from other buildings. It explores how knowledge obtained from one building can be transferred to another building for the purpose of energy forecasting.

Sensor-based forecasting techniques are numerous: a few examples are Support Vector Regression (SVR) [20], Neural Networks (NN) [21], gray-box models [22], decision trees [23], local learning [24], and multiple linear regression [25]. Ensemble approaches combine these techniques to improve prediction accuracy [26,27]. Several review studies have also been published: Yildiz et al. [28] reviewed commercial building electricity load forecasting, Deb et al. [29] surveyed time series forecasting techniques for building energy consumption, and Daut et al. [30] focused on conventional and artificial intelligence methods. Although these studies achieved excellent prediction accuracy in various context, they still did not support energy prediction for buildings with limited historical data. The present work builds on these studies by enabling transfer of forecasting knowledge among similar buildings.

The work of Mocanu et al. [31] focused on cross-building transfer learning and provided energy prediction solution for buildings with limited historical data by using data from other buildings. They combined reinforcement learning with a deep belief network to facilitate on-line learning. In contrast to Mocanu et al. who proposed a new energy forecasting approach, the present work is specifically looking at how forecasting knowledge from one building can be transferred to another building while continuing to use existing machine learning-based prediction techniques.

3.2. Transfer learning

Transfer learning can also be found in the literature under the term *cross-domain* followed by some other term such as learning [32], prediction [33], data [34], or data fusion [15]. Sometimes the term *domain* is replaced by the domain category (e.g., cross-company [34] and cross-building [31]).

Transfer learning has been recently used in many real-world problems: in software engineering, it has addressed cross-company software defect classification [33,35] and cross-company software effort estimation [34]; in voice processing, it has improved mispronunciation detection [36]; in image processing, it has dealt with visual recognition [32,37–42]; in natural language processing (NLP), it has addressed sentiment analysis [37,43–46]; in energy and buildings, it has improved energy [31] and temperature pre-

diction [47]. As discussed in Section 3.1, Mocanu et al. [31] focused on cross-building energy prediction. On the other hand Grubinger et al. [47] addressed temperature prediction: they merged the characteristics of the source(s) and target domains to produce a generalized domain that was used to predict the temperature in the target domain. Grubinger et al. used a customized prediction technique, unlike Hephaestus that can work with standard machine learning algorithms.

Although most transfer learning studies deal with classification tasks [32,33,35–41,43], regression tasks are mostly restricted to specific areas [34,42,44,46].

To manage transfer learning among domains, the domains must be adapted to an individual virtual domain using a procedure, which can be explicit or implicit, to reduce the distance between the allocated data in it. An implicit procedure processes the data in such way that reduces the distance as a consequence [43,44]. On the other hand, an explicit procedure transforms distance reduction into an optimization problem by directly attempting to minimize the distance, as presented in [33,45]. Both these methods rely on optimization algorithms, meaning that they are computationally expensive. Furthermore, they cannot work with other standard machine learning algorithms and have not been used for regression tasks, unlike Hephaestus that is designed to work with standard machine learning algorithms. As a compromise between implicit and explicit distance optimization, Li et al. [37] proposed heterogeneous domain adaptation to work with generic classification tasks. However, Hephaestus again has an advantage as it can work with all standard machine learning tasks, including regression and classification.

Although some of the studies mentioned at the beginning of this section naturally dealt with the time component (e.g., voice processing), none of them considered trends and seasonality (e.g., cross-building energy consumption). Microsoft SQL Server (a relational database management system) can use more than one source to create time series predictions, an ability that Microsoft calls *cross prediction* [48]. SQL Server embeds two algorithms: an autoregressive tree model with cross-predictions (ARTXP) [49] for short term predictions, and an autoregressive integrated moving average (ARIMA) for long term predictions. Cross-predictions are possible only for ARTXP models, and no information is publicly available about how it operates. However, it does not support features other than time.

Hephaestus deals with both time series and multi-feature regression together. It separates out the time component from the various domains, adapts the remainder component for all domains into a single domain, and uses any standard machine learning algorithm to create a predictive model.

4. Hephaestus method

This paper proposes Hephaestus, a novel cross-building energy prediction method based on transfer learning with seasonal and trend adjustment. To improve prediction for a target building, Hephaestus uses measurements from additional similar buildings collected over a much longer time frame. Moreover, Hephaestus is designed to work in the pre- and post-processing phases of standard machine learning, acting directly on top of feature and label values and without the need to modify the machine learning algorithm itself, making it possible to use any standard regression algorithm.

Hephaestus can be classified as an *inductive transfer learning* method. Considering that X represents the building feature space and the labels Y represent its energy consumption, it is clear that the predictive functions from the source and target domains are different ($P(Y_S|X_S) \neq P(Y_T|X_T)$) and therefore the learning tasks are also different ($\mathcal{T}_S \neq \mathcal{T}_T$).

Fig. 2 provides an overview of Hephaestus. The inputs are: (a) the *target*, which represents the target dataset and contains past information from the target building; (b) *sources 1 . n*, which represents additional datasets that will be used to improve the target prediction; and (c) the *predictive set*, which contains unlabeled data to be predicted.

Hephaestus consists of four main phases: (A) *time series adaptation*, in which the time effects (e.g. seasonality, trends) for all *source* and *target* datasets are analyzed, transferred to the target (if needed), and have seasonal and trend components removed; (B) *non-temporal domain adaptation*, in which the domains of non-temporal features and of the consumption data are adapted; (C) *standard machine learning*, which uses any appropriate standard algorithm to train the predictive model and generate predictions using the *predictive set*; and (D) *adjustment*, in which the prediction is readjusted using the factors calculated in the *time series adaptation* and *non-temporal domain adaptation* phases.

Hephaestus can be considered as both a parameter and an instance transfer method. It transfers the time factors and normalization parameters (if the target dataset does not have enough data to calculate them) as well as instances containing non-temporal features to train the predictive model. The following subsections discuss the four phases of Hephaestus.

4.1. Time series adaptation

Different buildings from the same category (e.g., schools, offices) can have distinct time profiles, but similar behavior for non-temporal features. For example, commercial building consumption profiles usually follow a weekly pattern, but may have peaks and valleys on different days of the week. Hence, it is important to remove seasonal and trend effects before analyzing and transferring the knowledge encoded by the correlation of non-temporal features and consumption data.

This phase has two objectives: (a) removing the effects of time from all datasets, and (b) transferring time series knowledge (if needed) from the sources to the target building. Fig. 3 illustrates this phase. The n raw source datasets and the target dataset are received as input. As output, the labels Y for each dataset contain the consumption data after trend and seasonality removal.

Formally, if each dataset has a unique time profile, the conditional probability $P(y|x_t)$, where x_t is a temporal feature (i.e., contains a value in the time domain), is not equal for all sources and the target. Therefore, the goal of *time series adaptation* $\Phi()$ here is to approximate the conditional probability $P(\Phi(Y)|X)$ of all buildings by removing the effects of all temporal features (e.g. day of the week) from consumption data.

4.1.1. Trend removal

The consumption data from each input dataset may have different trends. To reduce this difference, the *trend removal* procedure calculates the trend factor and removes it from the consumption data, which lose their long-term tendency, as can be seen in Fig. 3.

To calculate the trend factors, a two-step procedure is executed. First, *trend smoothing* is performed using moving averages:

$$ts_i = \frac{1}{m}\sum_{j=0}^{m-1} y_{i-j} \tag{5}$$

where ts_i is calculated for each data point i in the consumption time series and m represents the last m data points before i. In general, m is chosen to ensure that only long-term trends are removed. For instance, if the dataset consists of daily entries, m can be set to 365 to keep the cyclical components related to monthly and weekly seasonality.

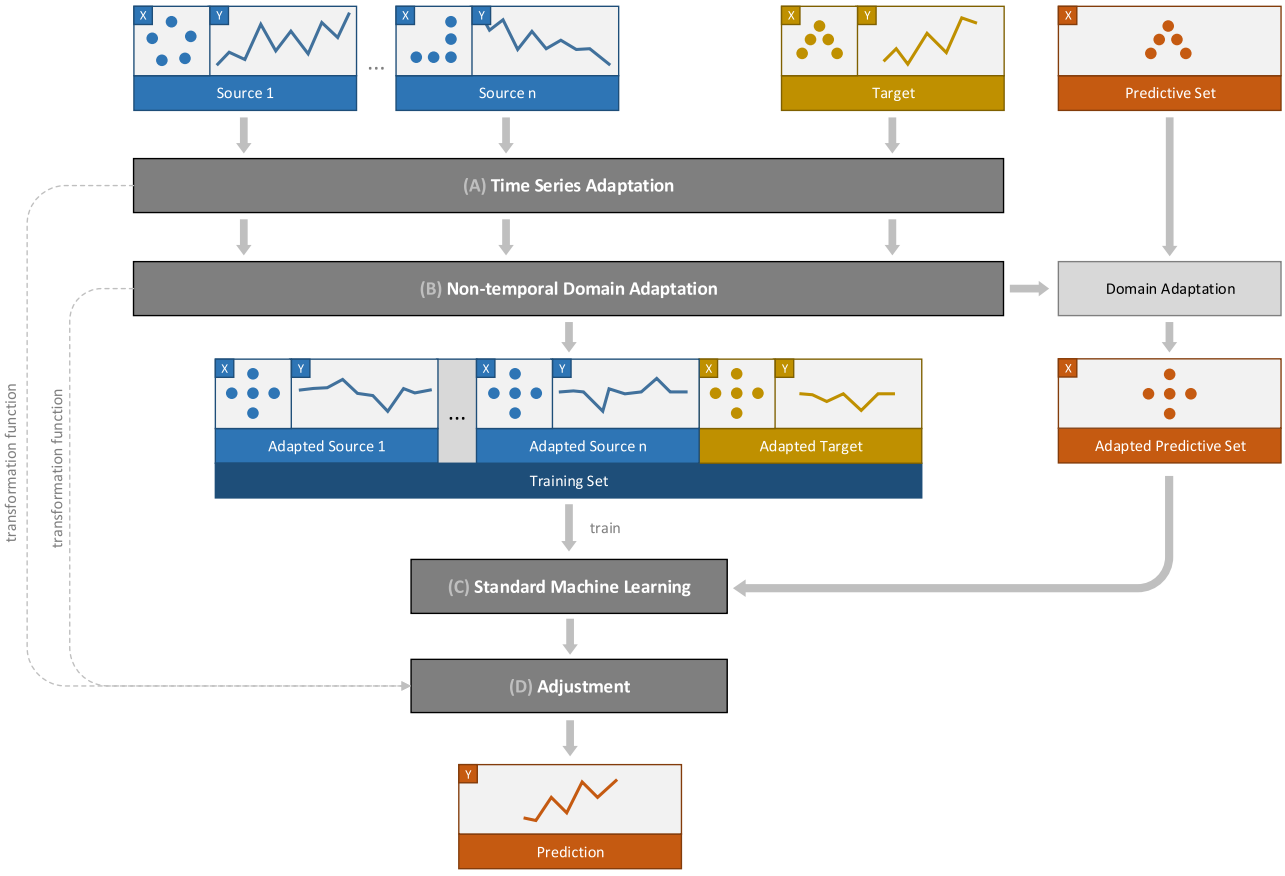


Fig. 2. Overview of Hephaestus method.

Fig. 3. Time series adaptation phase.

The second step involves *trend removal*. The trend factor is calculated by the equation:

$$tf_i = ts_i/ts_r \tag{6}$$

where r is the index of a reference value, commonly the last point of the dataset.

The calculated trend factors tf_i are used to remove the long term trend. If the time series follows the additive model described by Eq. (3), the trend is subtracted from the consumption data points, and if it follows the multiplicative model described by Eq. (4), the data points are divided by tf_i. Later, in the Adjustment phase (Section 4.4), the inverse procedure is executed to adjust the prediction.

Sometimes the small amount of data in the target dataset is statistically insufficient to determine the trend. This issue can be resolved by using the trend factors calculated from one of the source datasets or a composite of all of them, assuming that they have similar trends. For example, two approaches can be used: (a) calculating the average of all the source datasets or (b) choosing the dataset that is most similar to the target (e.g., using K-Nearest Neighbors).

4.1.2. Seasonality removal

Like trends, consumption data from different buildings can have different seasonal profiles (for example, weekly seasonality) that should be removed to reduce seasonal impact, as shown in Fig. 3. *Seasonality removal* calculates the seasonal indexes, first to remove them from the consumption data using a removal function, and later to adjust the prediction using the inverse of the removal function.

In this step, the same procedure is followed for each seasonality considered. Let P be the set of all values from a seasonality (e.g., days of the week), where p is a specific value in P, $y_{p,j}$ is the jth of m data points that occur in p, and $\bar{y}$ is the average of all data points from every $p \in P$. The seasonal index is calculated as:

$$s_p = \frac{1}{\bar{y}} \frac{1}{m} \sum_{j=1}^{m} y_{p,j} \qquad (p \in P) \tag{7}$$

This seasonal index s_p is then used to remove seasonality. If the time series follows the additive model described by Eq. (3), s_p is subtracted from the consumption data points, and if the consumption time series follows the multiplicative model described by Eq. (4), the consumption data points are divided by s_p.

As with trend removal, the target dataset may not be large enough to calculate seasonal indexes with statistical relevance. In this case, the target's seasonality removal can be calculated using the seasonal indexes from one of the source datasets or a composite of all of them, assuming that they have similar seasonal profiles.

The outcome of this phase, after trend and seasonality removal, is the residual component of consumption data with time effects removed.

4.2. Non-temporal domain adaptation

The goal of the *non-temporal domain adaptation* phase is to align the domains of all non-temporal features (those that do not define time and do not depend on it) and of the consumption data to enable them to be handled together. This paper proposes local and global normalization for the non-temporal domain adaptation, but Hephaestus is flexible enough to work with other domain adaptation techniques. Fig. 4 illustrates how this phase works.

For a feature f, the values X_f from each source dataset and the target dataset are processed using *global normalization* if the relationship between Y and X_f is absolute or using *local normalization* if this relationship is relative. These two normalization methods are described next.

4.2.1. Local normalization

A relationship between Y and X_f is relative if the value of Y relies on a proportional value of X_f. The conditional distributions after normalization $P(Y|\Psi(X_f))$ of each source and the target are the same, where $\Psi()$ is a normalization function. *Local normalization* is illustrated in Fig. 4, which shows all resulting marginal distributions centered in the same position for all source and target datasets.

The relation between energy consumption and external temperature is an example of a relative relationship. In this case, the effects of temperature on consumption are specific for each building because each has different heat-exchange characteristics with the outside environment. By normalizing temperature, these effects are aligned onto the same scale for all buildings.

Depending on the nature of the feature, min-max (Eq. (1)) or Z-score (Eq. (2)) normalization is used. In either case, only the feature values $X_f^{\mathcal{D}_k}$ from the dataset $\mathcal{D}_k$ belonging to the kth domain are used in the calculation. Fig. 4 uses z-score normalization as an example.

Similarly to time series adaptation, depending on target dataset size, it is sometimes difficult or statistically insufficient to achieve proper normalization using only the target dataset. This can be resolved by using the normalization parameters from one of the source datasets or a composite of all of them, assuming that they have similar distributions.

Because the consumption data from source and target datasets may follow different distributions, they must also be normalized. For example, energy consumption depends on building size: the energy consumption peak in a bigger building can be expected to be higher than in a smaller building. In this case, *local normalization* should be used.

4.2.2. Global normalization

A relationship between Y and X_f is absolute if the value of Y relies directly on the absolute value of X_f. In this case, the X_f of each dataset is a subset of a superset F that contains all subsets from all datasets, and the conditional distributions $P(Y|\Psi(X_f, F))$ for all source and target datasets are the same in a global context. Therefore, the feature X_f is considered to be already in the same domain for all buildings. *Global normalization* is illustrated in Fig. 4, where the dashed lines represent the assumed marginal distribution of the superset F and the continuous lines represent the marginal distribution of X_f for the source and target datasets.

For example, suppose that all the buildings are outfitted with the same type of equipment (e.g., computers) and that the number of pieces of equipment is a dataset feature. Assuming that each piece of equipment consumes the same amount of energy independently of where it is installed, its relationship with energy consumption would be considered absolute because the consumption depends directly on the number of pieces of equipment that are turned on.

Even though features with a global relationship are already in the same global domain, they should also be normalized to bring all features into the same scale and to ensure prediction quality. In this case, all the feature values X_f from all datasets are used in the normalization.

4.3. Standard machine learning

Hephaestus does not change how a machine learning (ML) algorithm works because the input to the ML algorithm is still one single dataset. The pre- and post-processing phases executed by Hephaestus do not affect the execution of the algorithm itself. Hence,

Fig. 4. Local and global z-score normalizations for relative and absolute relationships.

the method can work with any standard regression algorithm (e.g., support vector regression and neural networks) to build the predictive model, feeding the composed *training set* directly into the algorithm without the need to adapt the algorithm.

In addition, any other pre- or post-processing procedures, such as instance selection/weighting and feature selection/weighting, can be executed during the standard machine learning phase either before training or after prediction.

4.4. Adjustment

It is important to note that the labels (energy consumption data) have been modified during the *time series adaptation* and *non-temporal domain adaptation* phases. To achieve the correct predicted values, readjustments for each phase must be made by applying the inverse functions from the *time series adaptation* $\Phi^{-1}()$ and the *non-temporal domain adaptation* $\Psi^{-1}()$.

5. Case study: energy consumption

This case study aimed to use Hephaestus to improve energy consumption prediction for a building with only one month of data available, with the help of data from additional buildings. The goal was to reach similar accuracy as if the model had been trained with one entire year of data for the target building.

The evaluation was performed using data from four schools provided by Powersmiths [50], a Canadian company whose main product lines include meters and sensors to manage building resources. The schools are of different sizes and located across Newfoundland, Canada, which provides diverse weather conditions that are not too distinct from one another. Moreover, the schools share similar seasonal patterns, including hours of operations, holidays, and seasons of the year, but their weekly profiles are different from one another. The datasets for the four schools contain three years of daily data, from January 1, 2013, to December 31, 2015. Each dataset contains 17 attributes, including 4 temporal attributes: (1) year, (2) month, (3) day, (4) day of the week; 12 non-temporal attributes, all related to external weather: (5) minimum temperature, (6) maximum temperature, (7) mean temperature, (8) mean temperature difference from the day before, (9) mean temperature difference from two days before, (10) mean temperature difference from three days before, (11) dew point, (12) mean dew point, (13) minimum dew point, (14) minimum humidity, (15) maximum humidity, and (16) mean humidity; and finally the label (17) energy consumption, which is measured and collected by the Powersmiths meters.

Fig. 5 shows a histogram of the daily energy consumption for each school. The histograms clearly show that each school has a unique energy profile and therefore should be described by a distinct prediction function. In this context, the use of Hephaestus is essential to adapt the datasets and enable them to be used in conjunction.

In the following discussion, the *Evaluation* subsection describes how the performance of Hephaestus was evaluated; the *Implementation* subsection describes certain important implementation details; the *Preliminary Analysis* subsection examines details of the data used in the case study; and the *Results* subsection discusses how Hephaestus performed.

5.1. Evaluation

To evaluate the proposed approach, five prediction models were implemented for each school:

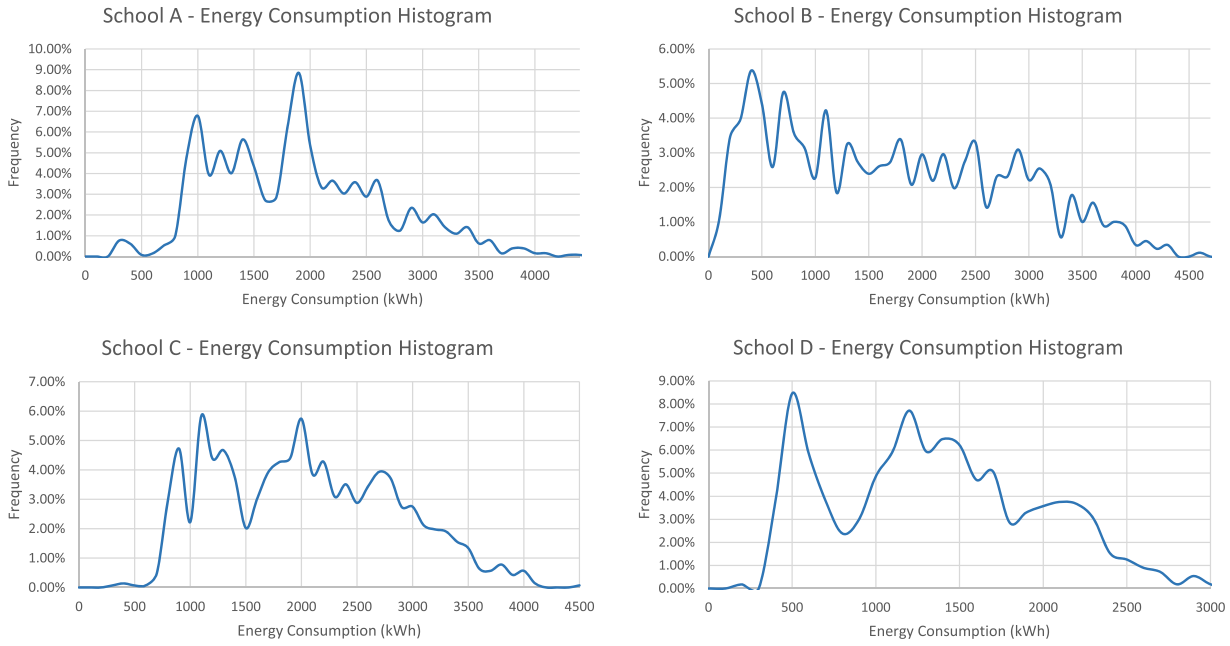


Fig. 5. Relative frequency histogram of energy consumption for each of the four schools.

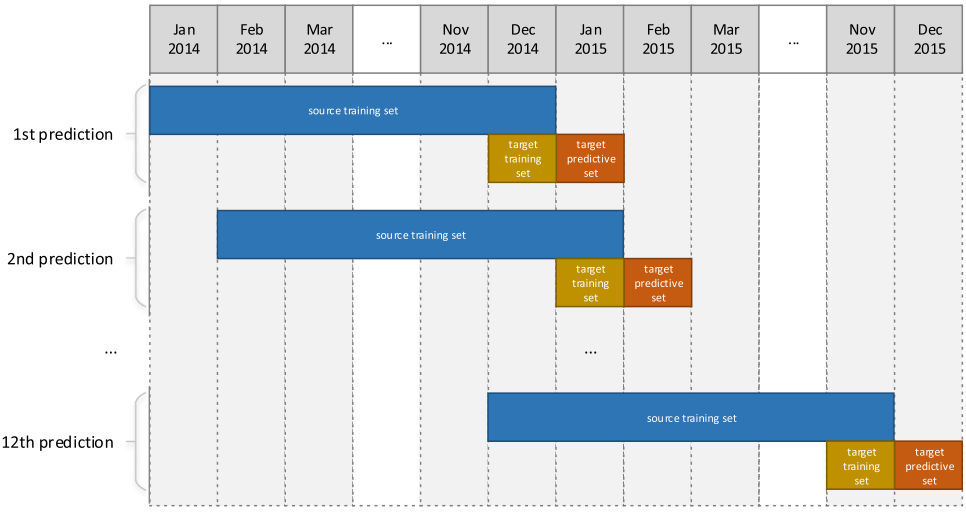


Fig. 6. Rolling base forecasting.

- **T.1**: ML algorithm (without Hephaestus) with one month of data from the target school, to simulate the scenario where only recent data from the target building are available.
- **T.12**: ML algorithm (without Hephaestus) with 12 months of data from the target school, used as a benchmark to compare with Hephaestus performance.
- **H.1-12**: Hephaestus with one month of data from the target school plus 12 months from the other schools.
- **H.12-12**: Hephaestus with 12 months of data from the target school plus 12 months of data from the other schools, used only as a benchmark.
- **N.1-12**: ML algorithm (without Hephaestus) with one month of data from the target school plus 12 months of data from the other schools. In this case, only min-max normalization was used for each feature to demonstrate the impact of Hephaestus.

Here *'the other schools'* refers to the three schools besides the target school itself. Hephaestus model H.1-12 was trained with data from the three schools, , and the prediction was evaluated on the fourth school. The predictions were calculated for one entire year using rolling base forecasting, which uses a certain number of months to train the model and one month to test. Fig. 6 illustrates how this process works, using as an example one month of data from the target school plus 12 months from the other schools.

The hypothesis to be tested was that Hephaestus models H.1-12 and H.12-12 should be better than model T.1 and similar to model T.12. Moreover, was is expected that model N.1–12 should not perform well because it used data from different domains to train the predictive model without any adaptation.

5.2. Implementation

The data for the analyzed schools showed that if the energy consumption is high its seasonal variations (differences between highs and lows) are also high. For example, Fig. 7 shows a sample of energy consumption for School C. The graph clearly demonstrates that both the energy consumption and its variation around trend tended to increase over time. Because of this pattern, this ex-

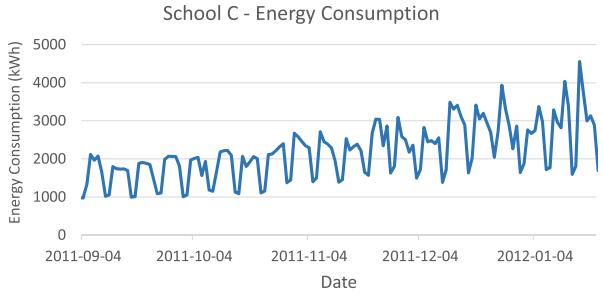


Fig. 7. Sample of an energy consumption timeline.

periment was implemented using the multiplicative model for the consumption time series.

Trend removal was performed according to Eqs. (5) and (6) with $m = 365$. By using a large m, the cyclical components related to seasonal weather features were kept. Using a smaller number of days would remove, for example, the correlation between temperature and energy consumption because the seasons of the year would be removed with the trend. For seasonal removal, only weekly seasonality was considered.

From the three years of data, the first year was used exclusively to calculate the long-term trend for the additional schools. The remaining two years were used to run the evaluation. Because all models predicted only one month ahead, calculating the trend for the target school was not necessary.

In the *non-temporal domain adaptation* phase, all the non-temporal attributes were normalized locally using z-scores (Eq. 2).

For the *standard machine learning* phase, two experiments were constructed using two algorithms (a) a multilayer perceptron (MLP) and (b) support vector regression (SVR). These algorithms were chosen because they are commonly used for energy consumption prediction [51]. This step was performed to demonstrate that Hephaestus works with different standard machine learning algorithms.

The *adjustment* was calculated by applying the inverse of the z-score function to the results from *standard machine learning* to readjust for non-temporal domain adaptation and by multiplying by the respective trend factor and seasonal index to reverse time series domain adaptation, a procedure that can be written as:

$$y = (z \times \sigma + \mu) \times tf \times s_p \tag{8}$$

5.3. Preliminary analysis

This section describes the details of the datasets used in this case study and helps to explain how Hephaestus works. Because the focus of this subsection is on analyzing data not on the prediction itself, this subsection used the entire dataset from all schools.

The weekly seasonal profiles for each school were calculated during *seasonal removal* in the *time series adaptation* phase. Fig. 8 confirms that the school's weekly profiles are different. Therefore, data from one school cannot be used directly to predict the energy consumption of another school because the prediction's seasonality will not fit the target's seasonality.

Furthermore, the correlation between non-temporal features (e.g. external mean temperature) and energy consumption is not the same for all schools. Fig. 9(a) shows the correlation between mean temperature and energy consumption, using a polynomial interpolation of degree 3, before *non-temporal domain adaptation*. In this scenario, determining the energy consumption of one school using the curve from another school is not accurate because the curves do not overlay each other. This demonstrates the need for domain adaptation, which brings these curves close together to improve machine learning accuracy.

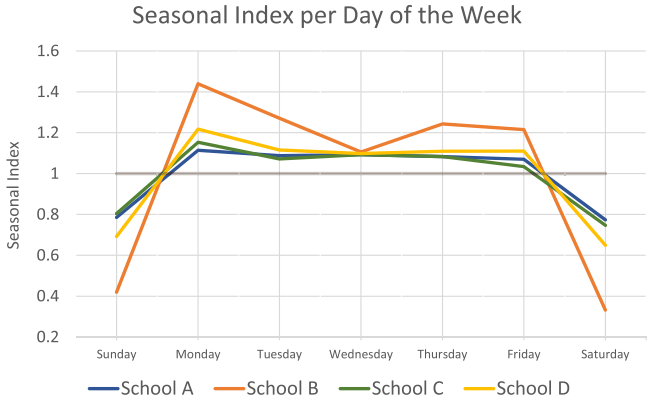
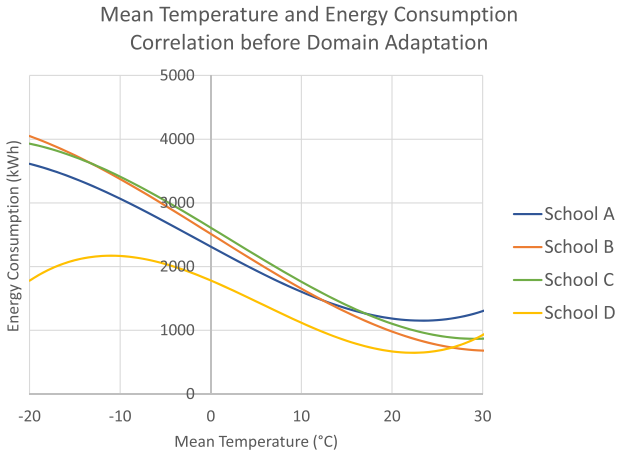
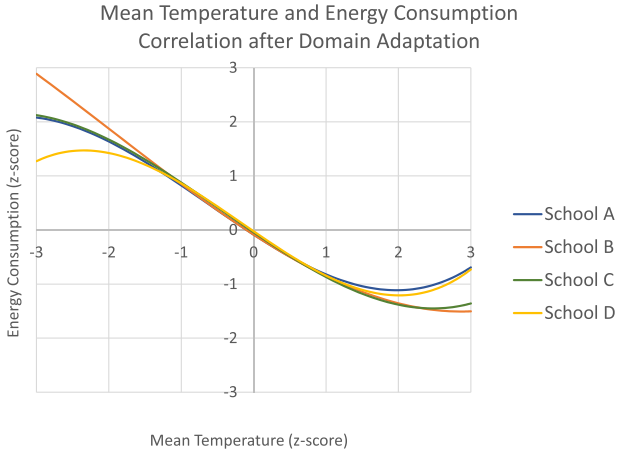


Fig. 8. Seasonal index for all four schools.



(a)



(b)

Fig. 9. Polynomial interpolation of the correlations between mean temperature and energy consumption.

Fig. 9(b) shows the correlation between external mean temperature and energy consumption after the *non-temporal domain adaptation* phase. It is clear that the procedure successfully reduced the distance between the data distributions for each school. Once the correlations for non-temporal features have been approximated, they are ready to feed into the standard machine learning algorithm.

Table 1
Energy consumption prediction errors for the schools (MLP).

Model	School A		School B		School C		School D	
	MAPE	MSE	MAPE	MSE	MAPE	MSE	MAPE	MSE
T.1	0.2088	229,548	0.3405	320,401	0.2105	264,078	0.3055	180463
T.12	0.1196	104,444	0.2281	132,927	0.1040	76,155	0.2237	118421
H.1–12	0.1191	86,240	0.2288	152,466	0.0986	74,601	0.2149	141240
H.12-12	0.1156	83,337	0.2284	162,093	0.0928	66,509	0.2060	123051
N.1–12	0.1928	277,025	0.6439	1,031,830	0.1941	234,497	0.2662	222852

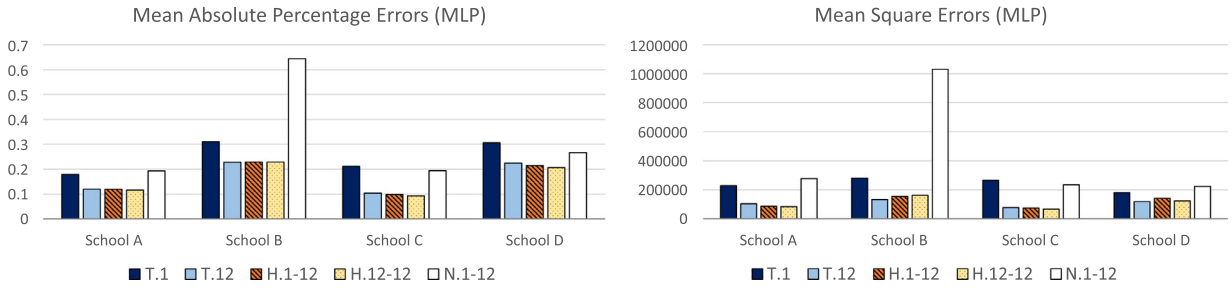


Fig. 10. Comparison of schools and models with their respective energy consumption prediction errors using MLP.

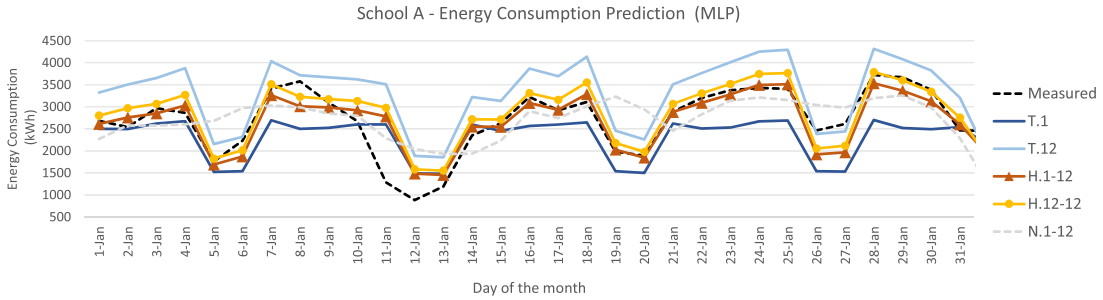


Fig. 11. Timeline comparison of predictions for MLP models.

5.4. Results

Table 1 shows the results of the experiment using the multilayer perceptron (MLP). MLP training can get stuck in a local minimum and therefore, MLP training was repeated ten times with different initial weights and the best model was selected for further evaluations. The columns Table 1 represent the Mean Absolute Percentage Error (MAPE) and the Mean Square Error (MSE) for each target school. MAPE expresses average absolute error as a percentage whereas MSE measures the average of the squares of the errors. MAPE and MSE are calculated as follows:

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y}_i|}{y_i} \tag{9}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{10}$$

where y_i is the actual consumption, $\hat{y}_i$ is the predicted consumption, and N is the number of observations.

These same values are plotted in Fig. 10. Fig. 11 shows a timeline chart comparing the predictions made by the various models using MLP.

Model N.1-12 proved to be the worst model for all schools. This was expected because this model used data that were outside the target domain, making wrong inferences and thus increasing the error. This shows that data from other schools cannot be used as-is, demonstrating the importance of the Hephaestus method.

Model T.1 had the second worst results for every school. By adding data from other schools using Hephaestus in model H.1-12, all schools improved their results compared with model T.1. When calculating T.1 minus H.1-12 from Table 1, school C had the highest improvement, reducing MAPE by 11.2% and MSE by 189,477.

In addition, the prediction accuracy for model H.1-12 was almost as good as for model T.12, which used 12 months from the target school. Moreover, H.1-12 performed even better than T.12 for schools A and C (when considering both MAPE and MSE) and D (when considering MAPE only).

Noted, however, that model T.12 gave better results than models H.1-12 and H.12-12 for schools B (considering MAPE and MSE) and D (considering only MSE). When the two metrics indicate different results, such as the school C MAPE value better for model H.1-12 than for model T.12 and the MSE value better for model T.12 than model H1.1-12 (Table 2), the conclusion cannot be drawn about which model performed better overall. However, the primary goal of Hephaestus is to improve prediction for buildings with small datasets, which was simulated by model T.1. Model T.12 was added only as a benchmark to verify whether H.1-12 could achieve accuracy similar to that obtained when 12 months of the target dataset were available, which is not possible in the problem Hephaestus was designed to solve.

Table 2 shows the results for support vector regression (SVR), which are plotted in Fig. 12. The results are similar to the MLP experiment. This demonstrates that Hephaestus can work with any standard machine learning algorithm.

Table 2
Energy consumption prediction errors for the schools (SVR).

Model	School A		School B		School C		School D	
	MAPE	MSE	MAPE	MSE	MAPE	MSE	MAPE	MSE
T.1	0.1798	185,051	0.2853	226,727	0.1594	160,584	0.2447	185,406
T.12	0.1327	132,285	0.2240	140,457	0.1006	76,340	0.2081	121938
H.1–12	0.1218	94,187	0.2272	142,427	0.0991	79,346	0.2227	156124
H.12-12	0.1133	85,407	0.2296	147,068	0.0940	70,158	0.2124	136681
N.1–12	0.1964	272,301	0.6623	986,109	0.2048	269,871	0.2656	224486

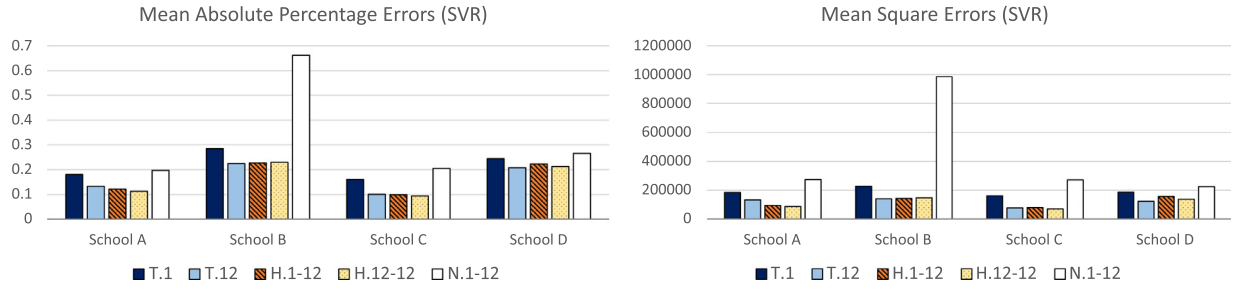


Fig. 12. Comparison of schools and models with their respective energy consumption prediction errors using SVR.

6. Conclusions

This paper has proposed Hephaestus, a novel cross-building energy prediction method based on transfer learning with seasonal and trend adjustment of time series. To improve predictions for a target building with a small data set, the proposed method uses measurements from other similar buildings collected over a much longer time frame. Hephaestus works in the pre- and post-processing phases, enabling the use of standard machine learning algorithms. The method adjusts the data from multiple buildings by removing the effects of time through time series adaptation and prepares time independent features through non-temporal domain adaptation.

To validate Hephaestus, a case study on energy consumption prediction for multiple schools was presented. Prediction accuracy increased by up to 11.2% using data from additional schools compared with a model that used only one month of data from the target school. Even with only one month's worth of data for the target school, the prediction accuracy was similar to or even better than that using 12 months of data from the target school

Future work will explore adding an instance weighting method to improve the performance of Hephaestus further. More experiments will be performed using a smaller number of data samples (i.e, days) from the target school to investigate how many data points are needed for the Hephaestus approach.

Acknowledgments

This research was supported in part by an NSERC CRD at Western University (CRDPJ 453294-13). In addition, the authors would like to acknowledge the support provided by Powersmiths.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.enbuild.2018.01.034.

References

[1] R.K. Jain, K.M. Smith, P.J. Culligan, J.E. Taylor, Forecasting energy consumption of multi-family residential buildings using support vector regression: investigating the impact of temporal and spatial monitoring granularity on performance accuracy, Appl. Energy 123 (2014) 168–178.
[2] M. Ghofrani, M. Ghayekhloo, A. Arabali, A. Ghayekhloo, A hybrid short-term load forecasting with a new input selection framework, Energy 81 (2015) 777–786.
[3] R.J. Hyndman, A.V. Kostenko, Minimum sample size requirements for seasonal forecasting models, Foresight 6 (2007) 12–15.
[4] S.B. Green, How many subjects does it take to do a regression analysis, Multivariate Behav. Res. 26 (3) (1991) 499–510.
[5] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.
[6] D. Pyle, Data Preparation for Data Mining, Morgan Kaufmann, 1999.
[7] P.M. Kroonenberg, Applied Multiway Data Analysis, Wiley, 2008.
[8] P.C. Bruce, Introductory Statistics and Analytics: A Resampling Perspective, Wiley, 2014.
[9] J. Qifa, A solution to seasonal adjustment forecasting for hydraulic smes in investment decision-making, in: Proceedings of the Control and Decision Conference (2014 CCDC), IEEE, Changsha, China, 2014, pp. 724–729.
[10] W.C. Hong, Electric load forecasting by seasonal recurrent svr (support vector regression) with chaotic artificial bee colony algorithm, Energy 36 (9) (2011) 5568–5578.
[11] J. Wang, W. Zhu, W. Zhang, D. Sun, A trend fixed on firstly and seasonal adjustment model combined with the ε-svr for short-term forecasting of electricity demand, Energy Policy 37 (11) (2009) 4901–4909.
[12] S. Hylleberg, Seasonality in Regression, Academic Press, 1986.
[13] R.J. Hyndman, G. Athanasopoulos, Forecasting: Principles and Practice, OTexts, 2014.
[14] A.L. Heureux, K. Grolinger, H.F. ElYamany, M.A.M. Capretz, Machine learning with big data: challenges and approaches, IEEE Access 5 (2017) 7776–7797.
[15] Y. Zheng, Methodologies for cross-domain data fusion: an overview, IEEE Trans. Big Data 1 (1) (2015) 16–34.
[16] Y.T. Chae, R. Horesh, Y. Hwang, Y.M. Lee, Artificial neural network model for forecasting sub-hourly electricity usage in commercial buildings, Energy Build. 111 (2016) 184–194.
[17] A. Tascikaraoglu, B.M. Sanandaji, Short-term residential electric load forecasting: a compressive spatio-temporal approach, Energy Build. 111 (2016) 380–392.
[18] C. Deb, L.S. Eang, J. Yang, M. Santamouris, Forecasting diurnal cooling energy load for institutional buildings using artificial neural networks, Energy Build. 121 (2016) 284–297.
[19] K. Grolinger, A.L. Heureux, M.A. Capretz, L. Seewald, Energy forecasting for event venues: big data and prediction accuracy, Energy Build. 112 (2016) 222–233.
[20] Y. Chen, P. Xu, Y. Chu, W. Li, Y. Wu, L. Ni, Y. Bao, K. Wang, Short-term electrical load forecasting using the support vector regression (svr) model to calculate the demand response baseline for office buildings, Appl. Energy 195 (2017) 659–670.
[21] L. Xiao, W. Shao, M. Yu, J. Ma, C. Jin, Research and application of a hybrid wavelet neural network model with the improved cuckoo search algorithm for electrical power system forecasting, Appl. Energy 198 (2017) 203–222.
[22] H. Harb, N. Boyanov, L. Hernandez, R. Streblow, D. Müller, Development and validation of grey-box models for forecasting the thermal response of occupied buildings, Energy Build. 117 (2016) 199–207.
[23] Z. Yu, F. Haghighat, B.C. Fung, H. Yoshino, A decision tree method for building energy demand modeling, Energy Build. 42 (10) (2010) 1637–1646.
[24] K. Grolinger, M.A. Capretz, L. Seewald, Energy consumption prediction with big data: balancing prediction accuracy and computational resources, in: Proceedings of the IEEE International Congress on Big Data (BigData Congress), 2016, pp. 157–164.

[25] T. Fang, R. Lahdelma, Evaluation of a multiple linear regression model and sarima model in forecasting heat demand for district heating system, Appl. Energy 179 (2016) 544–552.
[26] J.S. Chou, D.K. Bui, Modeling heating and cooling loads by artificial intelligence for energy-efficient building design, Energy Build. 82 (2014) 437–446.
[27] D.B. Araya, K. Grolinger, H.F. ElYamany, M.A. Capretz, G. Bitsuamlak, An ensemble learning framework for anomaly detection in building energy consumption, Energy Build. 144 (2017) 191–206.
[28] B. Yildiz, J. Bilbao, A. Sproul, A review and analysis of regression and machine learning models on commercial building electricity load forecasting, Renewable Sustainable Energy Rev. 73 (2017) 1104–1122.
[29] C. Deb, F. Zhang, J. Yang, S.E. Lee, K.W. Shah, A review on time series forecasting techniques for building energy consumption, Renewable Sustainable Energy Rev. 74 (2017) 902–924.
[30] M.A.M. Daut, M.Y. Hassan, H. Abdullah, H.A. Rahman, M.P. Abdullah, F. Hussin, Building electrical energy consumption forecasting analysis using conventional and artificial intelligence methods: a review, Renewable Sustainable Energy Rev. 70 (2016) 1108–1118.
[31] E. Mocanu, P.H. Nguyen, W.L. Kling, M. Gibescu, Unsupervised energy prediction in a smart grid context using reinforcement cross-building transfer learning, Energy Build. 116 (2016) 646–655.
[32] F. Zhu, L. Shao, Weakly-supervised cross-domain dictionary learning for visual recognition, Int. J. Comput. Vision 109 (1–2) (2014) 42–59.
[33] Y. Ma, G. Luo, X. Zeng, A. Chen, Transfer learning for cross-company software defect prediction, Inf. Softw. Technol. 54 (3) (2012) 248–256.
[34] L.L. Minku, X. Yao, How to make best use of cross-company data in software effort estimation? in: Proceedings of the 36th International Conference on Software Engineering, ACM, New York, NY, 2014, pp. 446–456.
[35] J. Nam, S.J. Pan, S. Kim, Transfer defect learning, in: Proceedings of the 2013 International Conference on Software Engineering, IEEE Press, San Francisco, CA, 2013.
[36] W. Hu, Y. Qian, F.K. Soong, Y. Wang, Improved mispronunciation detection with deep neural network trained acoustic models and transfer learning based logistic regression classifiers, Speech Commun. 67 (2015) 154–166.
[37] W. Li, L. Duan, D. Xu, I.W. Tsang, Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation, IEEE Trans. Pattern Anal. Mach. Intell. 36 (6) (2014) 1134–1148.
[38] H. Hosseinzadeh, F. Razzazi, E. Kabir, A weakly supervised large margin domain adaptation method for isolated handwritten digit recognition, J. Visual Commun. Image Represent. 38 (2016) 307–315.
[39] A.S. Mozafari, M. Jamzad, A svm-based model-transferring method for heterogeneous domain adaptation, Pattern Recognit. 56 (2016) 142–158.
[40] B. Gong, Y. Shi, F. Sha, K. Grauman, Geodesic flow kernel for unsupervised domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, 2012, pp. 2066–2073.
[41] V.M. Patel, R. Gopalan, R. Li, R. Chellappa, Visual domain adaptation: a survey of recent advances, IEEE Signal Process. Mag. 32 (3) (2015) 53–69.
[42] M. Yamada, L. Sigal, Y. Chang, Domain adaptation for structured regression, Int. J. Comput. Vision 109 (1–2) (2014) 126–145.
[43] X. Hu, J. Pan, P. Li, H. Li, W. He, Y. Zhang, Multi-bridge transfer learning, Knowl. Based Syst. 97 (2016) 60–74.
[44] S.J. Pan, I.W. Tsang, J.T. Kwok, Q. Yang, Domain adaptation via transfer component analysis, IEEE Trans. Neural Netw. 22 (2) (2011) 199–210.
[45] H. Daumé III, Frustratingly easy domain adaptation, in: Proceedings of the 45th Annual Meeting, Association for Computational Linguistics, Prague, Czech Republic, 2007, pp. 256–263.
[46] C. Cortes, M. Mohri, Domain adaptation and sample bias correction theory and algorithm for regression, Theor. Comput. Sci. 519 (2014) 103–126.
[47] T. Grubinger, G.C. Chasparis, T. Natschläger, Generalized online transfer learning for climate control in residential buildings, Energy Build. 139 (2017) 63–71.
[48] Microsoft time series algorithm technical reference, Accessed: 06-04-2016. https://msdn.microsoft.com/en-us/library/bb677216.aspx.
[49] C. Meek, D.M. Chickering, D. Heckerman, Autoregressive tree models for time-series analysis, in: Proceedings of the 2nd International SIAM Conference on Data Mining, SDM, Arlington, VA, 2002, pp. 229–244.
[50] Powersmiths international corp., Accessed: 07-24-2017. http://www.powersmiths.com/.
[51] a. S. Ahmad, M.Y. Hassan, M.P. Abdullah, H.a. Rahman, F. Hussin, H. Abdullah, R. Saidur, A review on applications of ANN and SVM for building electrical energy consumption forecasting, Renewable Sustainable Energy Rev. 33 (2014) 102–109.