August 14, 2016

Improving Demand Prediction in Bike Sharing System by Learning Global Features

Ming Zeng, *Carnegie Mellon University*
Tong Yu, *Carnegie Mellon University*
Xiao Wang, *Carnegie Mellon University*
Vincent Su, *Carnegie Mellon University*
Le T Nguyen, *Carnegie Mellon University*, et al.

Improving Demand Prediction in Bike Sharing System by Learning Global Features

Ming Zeng, Tong Yu, Xiao Wang, Vincent Su, Le T. Nguyen, Ole J. Mengshoel
{ming.zeng,tong.yu,sean.wang}@sv.cmu.edu
{vincent.su,le.nguyen,ole.mengshoel}@sv.cmu.edu
Carnegie Mellon University
NASA Research Park, Bldg 23
Moffett Field, CA 94035

ABSTRACT

A bike sharing system deploys bicycles at many open docking stations and makes them available to the public for shared use. These bikes can be checked-in and checked-out at any of the docking stations. Predicting daily visits is important for service providers to optimize bike allocation and station maintenance. In this paper, we formulate this prediction problem as a regression task. Through data analysis, we develop several features that are very helpful in predictions. Moreover, we demonstrate that there are significant differences among the patterns of visits at different stations. To improve prediction accuracy, we propose station-centric augmented with global feature transformation. The gradient boosting decision tree (GBDT) and neural network (NN) techniques are leveraged to extract global features. The experimental results demonstrate that the proposed model offers better prediction performance compared to two baseline approaches.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning

Keywords

Machine Learning; Prediction; Feature Transformation; On-demand Marketplace; Data Visualization

1. INTRODUCTION

Bike sharing systems have become more and more popular in recent years. Although there are various transportation options in urban areas, such as subways, buses and light rails, bike sharing systems can effectively connect these different transportation modes and provide people with more diverse choices in their everyday commute.

The benefits of predicting daily demand in a bike sharing system are manifold. With accurate prediction of the demand, service providers can perform effective and efficient maintenance of the station facilities since they will have a better estimation about when to repair or replace the bike docks or bikes. Meanwhile, for bike stations with a large number of visits every day, providing more bike docks can improve service quality and user experience. Furthermore, good predictions of the bike demand can help service providers in setting the price of service.

In this paper, our goal is to build a demand predictor to forecast the bike demand in a bike sharing system. The dataset used in our study is from New York City (NYC) - Citi Bike [1]. In this paper, the demand is defined as the daily number of visits to a station. [2] A bike can be checked out or returned at any given time from or to any station. We formulate our task as a regression problem, where the outcome to predict is a continuous variable. Considering various impacting factors such as weather, user activity and season, it is non-trivial to predict the daily demand of bikes.

Through data analysis, we engineer several useful features for demand forecasting. But data analysis also reveals an important problem: sometimes there exists significant differences between the patterns in data from different stations. Therefore, training a model on all labeled data may not be the optimal strategy as it does not take into account the differences. To improve prediction performance for a particular station, we study how to effectively make use of all labeled data and labeled data belonging to this bike station.

Motivated by the aforementioned challenge, we examine three methods: *city-centric model*, *station-centric model*, and *station-centric model with global features*. The *city-centric model* is a single predictive model using all the available training data from all stations. The *station-centric model* is individual predictive model for each station. We demonstrate the limitations of both the city-centric model and the station-centric model. Then, we propose a *station-centric model with global features*, where feature extraction models are trained on the entire dataset to learn extra features to capture global information. This hybrid approach can improve the accuracy of bike demand prediction.

Key contributions of this paper are as the following:

- We conduct data analysis for this Citi Bike demand prediction problem. We engineer several meaningful

DOI: 10.1145/1235

[1] https://www.citibikenyc.com/system-data

[2] In this paper, *demand* is used interchangeably with *visits*. This indicates how frequently users use the facilities, such as the bike docks and bikes in the bike sharing system.

features that make good predictions with regression models.

- We reveal significant variation in the demand patterns for different bike stations. Then, we discuss the limitation of two baseline approaches: city-centric model and station-centric model.
- We propose a hybrid station-centric model with global features transformation. We study and compare two methods of global feature extraction: gradient boosting decision tree (GBDT) and neural network (NN).

This paper is structured as follows. In Section 2, we briefly review related research. In Section 3, we describe the NYC Citi Bike dataset and discuss feature engineering. Our data analysis reveals greatly varying data patterns for different bike stations. Section 4 first presents two baseline methods and then our station-centric model with global features. In Section 5, we present experimental results to validate our method and discuss the merits and potential limitations of different approaches, which give us guidance on when to use which model in practice. Section 6 concludes and outlines future research directions.

2. RELATED RESEARCH

This section summarizes research work on demand prediction and the station-centric model with global features extraction.

2.1 Demand Prediction

Previous work [23] analyzes bike trips in the dataset from September 2015. It studies the number of bike trips taken per month, the average number of bike trips taken per hour on weekdays and weekends, the impact of age and gender on the number of trips, the percentage of bikes moved manually from one station to another in each area of NYC, and the impact of weather conditions on the average Citi bike trips taken per day. The paper presents a prediction model for Citi bike monthly trips using non-linear regression. It analyzes how the average number of bike trips varied with respect to several factors. However, as the analysis is city-centric approach, it may miss some unique station information that is important for individual stations.

The pairwise bike demand in rush hours is predicted using a linear regression model [25]. The taxi usage, weather and spatial factors are considered in the model. This model can help decision-makers to determine in which neighborhoods to expand their network.

To focus on system rebalancing and optimization problems [20, 21], although the authors attempt to find a solution to obtain daily demand prediction at various stations during rush hours, they estimate the lower bound of the true demand by simply averaging the number of trips for each time window.

To studies the bike rebalancing problem[18]. This work also studies bike pick-up and drop-off predictions. The pick-up demand is estimated by the weighted KNN approach. The weights are the similarities of weather, temperature and other environmental factors. The drop-off demand is predicted by estimating the arrival probability of pick-up bikes.

More broadly, there are several other works on demand prediction in other scenarios, such as railway demand forecasting [28], user demand in electronic commerce [8] and passenger demand prediction for bus services [12, 33].

2.2 Station-Centric Model and Feature Extraction

It is straightforward to build a single predictive model for all stations. However, the stations' features may be diverse and heterogeneous. Therefore, a single city-centric or "global" predictive model may not provide satisfying performance for individual stations. An alternative approach is to build a station-centric or "personalized" predictive model for each station. This "personalized" idea is widely used in many fields, for example in search engines [30], click prediction [5], weather forecasting [9], and health-care applications [13].

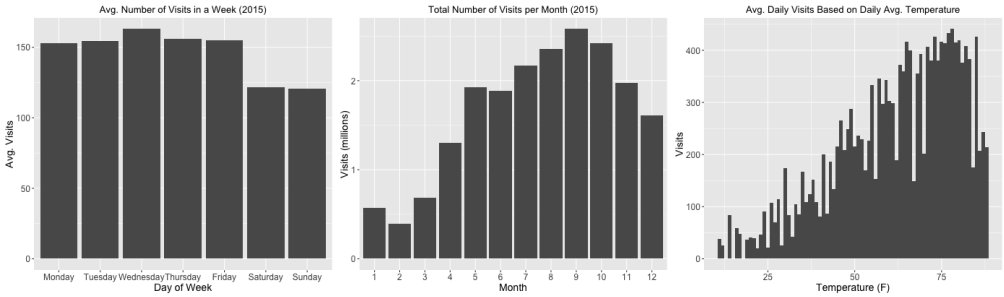
Choosing relevant features is important in the task [3], but when there is not enough data for a station, we may be not able to have a accurate predictive model. To alleviate this problem, hierarchical models allow us to leverage both "global" information and "personalized" information simultaneously [22, 15]. At one level, we calculate the visits for the entire stations. At another level, we calculate the station-specific demands. The less information we have about a particular station, the more closely it will approximate the across-station mean. Another solution is modeling similar stations demands, which can significantly improve demand prediction using the city-centric model [25].

Constructing features across all stations can incorporate global information into the model, which can improve the performance predictive model. The effectiveness of decision tree feature extraction is illustrated in Ads clicks prediction [10]. Typically, there are two simple effective methods for this transformation. For continuous features, the feature transformation is performed by binning the basic features and treating each bin index as a categorical feature. The linear classifier is able to successfully learn a non-linear map for the features. For this method to perform well, it is crucial to learn effective bin boundaries. The second transformation approach builds tuple input features. For categorical features, the naive brute force approach considers all possible combinations over basic features by taking the Cartesian product. If the basic features are continuous, we can perform joint binning using for example a k-d tree.

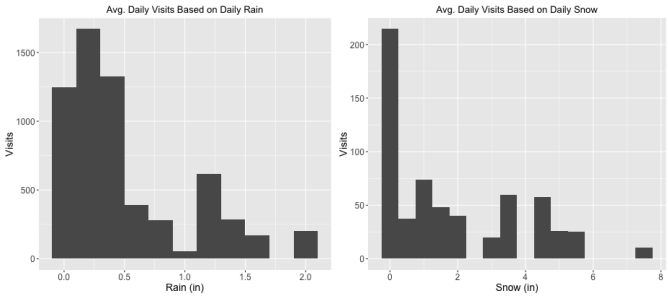
Neural networks have been successfully applied in feature extraction in many applications such as speech recognition [1], computer vision [24], and natural language processing [7, 14]. If the data is highly nonlinear, we can include multiple hidden layers in the NN creating a deep neural networks (DNN). The NNs can be trained in an unsupervised or a supervised manner. The unsupervised learning approaches include auto-encoder [2, 29], and Restricted Boltzmann machine (RBM) [11]. For supervised learning, the convolutional neural networks (CNN) [16] and recurrent neural networks (RNN) have achieved high performance in image processing [31], speech recognition tasks [6], and human activity recognition [32].

3. DATA ANALYSIS

In this section, we first describe the Citi Bike sharing dataset. Then we discuss our analysis of the dataset and as used as our engineering of features.



(a) Visits vs. day of week (b) Visits vs. month (c) Visits vs. temperature



(d) Visits vs. rain (e) Visits vs. snow

Figure 1: Number of visits (on y-axis) as a function of (a) day of week, (b) month, (c) temperature, (d) rain, (e) snow. The average visit is taken over all the NYC Citi Bike stations

3.1 The Dataset

The Citi Bike data gives information on every trip taken within a period of time. Started in September 2011, total distance traveled with Citi Bike already exceeded 20 million miles by 2014. Currently, there are approximately 1 million trips taken on average per month. The available fields of the dataset include Trip Duration (seconds), Start Time and Date, Stop Time and Date, Start Station Name, End Station Name, Station ID, Station Latitude and Longitude, Bike ID, User Type and Gender.

Our task is to extract useful features from these attributes and find out the ones with good predicting power of demand. Intuitively, time of day and week, weather, and events happening in the city would have large impact on the demand at the Citi Bike stations. Therefore, we also acquired NYC daily weather data to accompany our bike-share dataset, and extracted time information from the original dataset.

3.2 Feature Observations

First, we examine the relationship between day of week and demand in terms of number of visits in Figure 1a. We observe that demand on weekdays is on average about 25% higher than on weekends. Also, it peaks on Wednesdays. This can be explained by people's bike-to-work pattern in New York city.

Next, we examine total demand of Citi Bike in every month of 2015, depicted by Figure 1b. The curve indicates that there is more demand in summer while much less in winter. The dramatic difference between peak-to-bottom, or even just peak-to-average ratio is important for resource allocation.

Figure 1c shows demand as a function of temperature. As expected and seen from the previous figure, higher temperature tends to drive demand higher. What is interesting in this graph is the flatten part on both ends. These bends imply that when the temperature is extremely high or extremely low, its fluctuation does not really change demand too much. In contrast, with mild temperature, little fluctuation can result in very different demand levels.

The next two figures show us the relationship between demand and rain, snow, respectively. In Figure 1d, we observe that days with low or no precipitation have a variety of demand levels since the demand might be affected by other factors. We also see that in presence of heavy rain, there is still a good amount of demand. Unlike rain levels, Figure 1e shows a clear relationship between demand and the amount of snow. The demand drops drastically when it starts from no snow to moderate amount of snow, and continues to drop slowly as snow gets thicker.

The aforementioned observations give us some understandings of our data, as well as what kind of features can be good candidates for our machine learning model to predict the demand level.

3.3 Different Data Patterns in Different Stations

In this section, we compare the data of different stations. In some cases, there are significant differences between different stations.

Some representative examples are shown in Figure 2. In Figure 2a and 2b, we show that station 72, 83 and 150 exhibit significant difference in data distribution. The x-axis is the feature of min temperature and y-axis is the bike demand. If we train a model on the data from Station 72 and test it on the data from Stations 150, the prediction could be very inaccurate. Similar situations apply to other features such

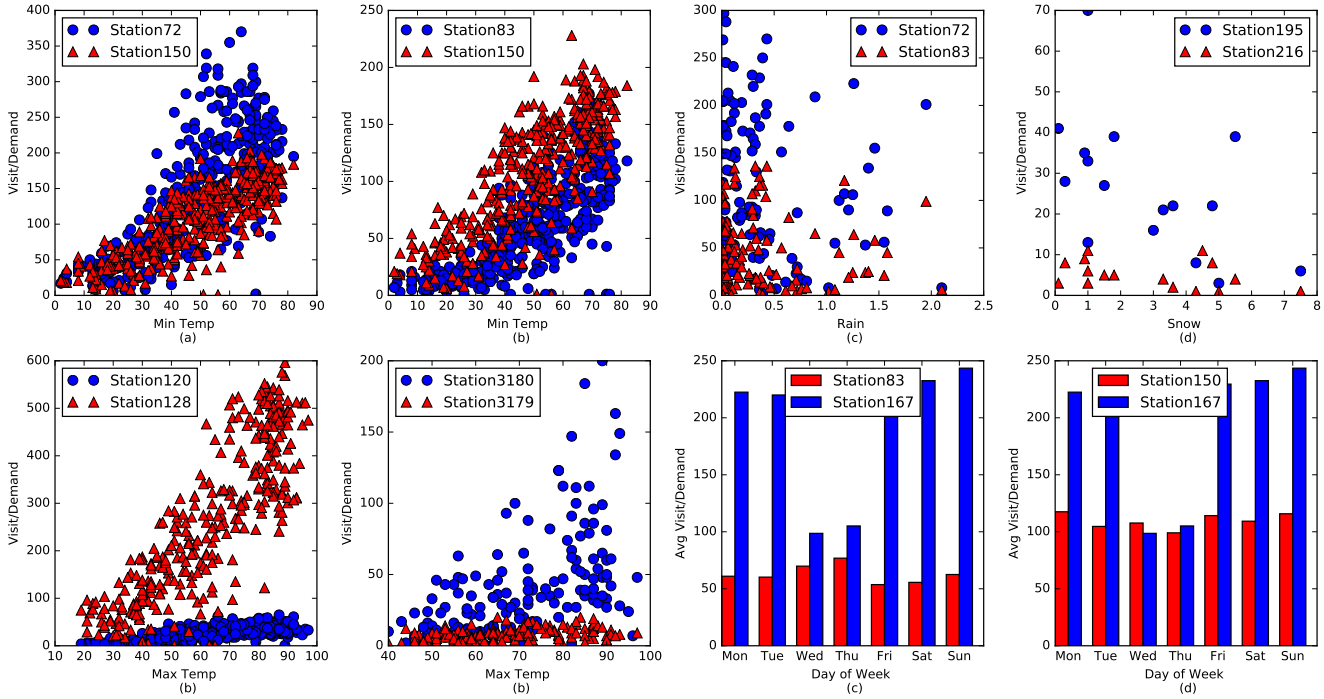


Figure 2: Some representative examples showing the difference between the data distribution of different stations. The x-axis shows the feature and the y-axis indicates the value of each sample.

ID	Feature Description
$f_1, \cdots, f_7$	Day of week by one-hot encoding
f_8	Holiday or not
f_9	Season
f_{10}	Max temperature
f_{11}	Min temperature
f_{12}	Precipitation
f_{13}	Snow (inch)
f_{14}	Rainfall
$f_{15}, \cdots, f_{21}$	7 days historical demand

Table 1: The engineered features as the input of city-centric model and station-centric model.

as rain, snow, max temperature, and day of the week as seen in Figure 2.

These observations indicate that if we want to predict the bike demand for a particular station, it may not be the optimal strategy to train a model on the labeled data from all stations. This motivates our studies in Section 4.

4. A STATION-CENTRIC MODEL WITH GLOBAL FEATURES

Motivated by the different data patterns for different bike stations and the introduction of new bike stations, in this section we present a city-centric model as a baseline, a station-centric model, and our station-centric model with global feature transformation.

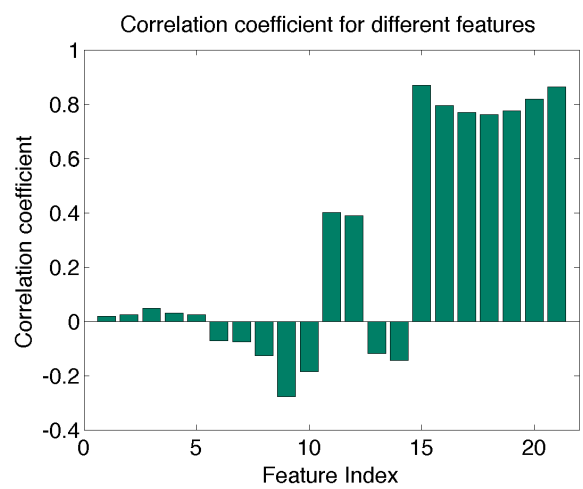


Figure 3: The correlation coefficients between each feature in Table 1 and the demand.

4.1 Motivations

A trivial way is to train a model on the dataset from all stations. This approach is referred as the city-centric model, which is discussed in Section 4.2. As discussed in Section 3.3, we observe significant distribution differences between various stations. It means that when we predict the demand for some stations, we may train the model on the data with different distributions, which will violate the i.i.d assumption in machine learning algorithms.

At the same time, in the empirical data we find that the new bike stations steadily increase in 2015, as shown in Fig-

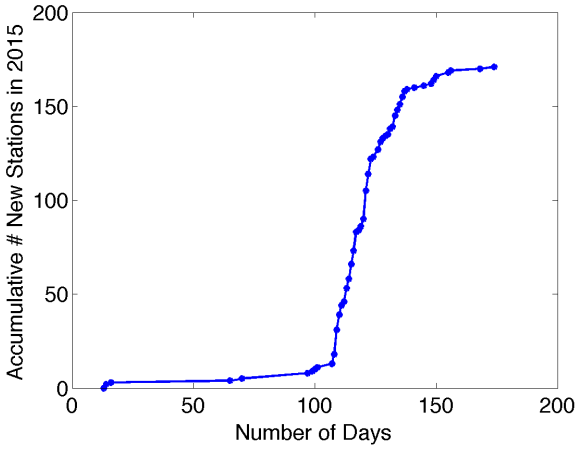


Figure 4: The number of new stations deployed in the days of year 2015. In 2015, the new bike stations are kept being deployed in New York city. The total number of bike stations steadily increases, with a dramatic increase between days 110 and 140.

ure 4. For each new station, when it is deployed, there is no labeled training data available.

The data patterns between different stations are not identical, and new bike stations with limited training data are common. Those two facts motivate us to make good use of the available training dataset.

4.2 City-Centric Model

In the city-centric model, we aggregate all the available training data across different bike stations in the city. Then we build a single regression model using all the training data.

The regression algorithms used in training the city-centric model: linear regression (LR), support vector regression (SVR) [27], decision tree (DT) and random forest (RF) with 1000 trees [4]. To be consistent and allow comparison, the four algorithms are also used in the station-centric model in Section 4.3 and the station-centric model with global feature transformation in Section 4.4.

4.3 Station-Centric Model

Because the feature distributions of different stations are heterogeneous, the "one size fits all" city-centric model may not be the best for all stations. Instead of using the training data in a city-centralized fashion, the station-centric model is built for individual stations. So the number of regression models is the same as the number of the stations. In the station-centric model, the training data and the test data have the same distribution, so the regression model can more effectively capture the localized information and provide better demand predictions.

We also study linear regression, support vector regression, decision tree and random forest the prediction task in the station-centric model.

In this approach, the amount of training data would be much smaller compared with the city-centric model in Section 4.2. As a result, this trained model may be fragile, because occasionally some test data with special patterns may not be captured in the relative small training set for the station-centric model.

4.4 Station-Centric Model with Global Feature Transformation Framework

In order to overcome the potential issue of station-centric model in Section 4.3, we try to capture global information from the entire dataset.

We now consider extracting additional features to improve the prediction performance. The feature extraction model (Fig 6) is trained on the entire dataset. Then the new features are transformed from the trained feature extraction model to maintain the data global information. We introduce two feature transformation approaches in this section: decision tree feature transformation and neural network feature transformation.

4.4.1 Decision tree feature transformation

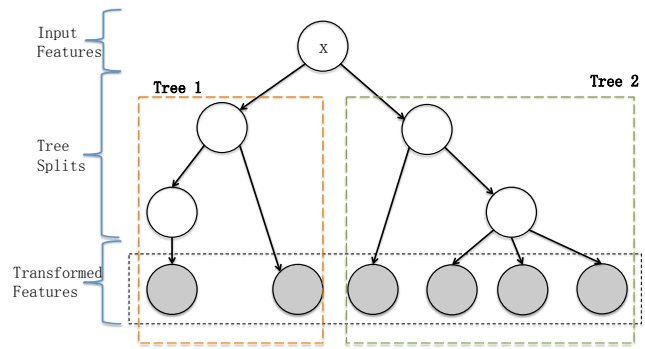


Figure 5: Input features are transformed by the boosted decision tree model. Boosted decision trees prove to be very useful for feature transformation [10].

The boosted decision tree can serve as a powerful tool to implement the aforementioned non-linear and tuple transformations [10]. In practice, we treat each individual tree as a categorical feature. Its categorical value is the index of the leaf node which an instance ends up falling in. We use the traditional *1-of-k* coding for this type of categorical features. For instance, consider the boosted tree model in Figure 5 with 2 subtrees, where "Tree 1" has two leaf nodes and the "Tree 2" has four leaf nodes. If an instance ends up in leaf 1 in the first subtree and leaf 2 in second subtree, the final output of the transformation and the subsequent input to the linear classifier will be a binary vector $[1, 0, 0, 1, 0, 0]$, where the first two entries correspond to the leaf nodes of the first subtree and last four to those of the second subtree. In each learning iteration, a new tree is created to model the residual of previous trees. We can understand boosted decision tree based transformation as a supervised feature encoding that converts a real-valued vector into a compact binary-valued vector. A traversal from root node to a leaf node represents a rule on certain features. Fitting a linear classifier on the binary vector is essentially learning weights for the set of rules.

4.4.2 Neural network feature transformation

In NN, while classification typically uses a form of logistic regression/softmax in the last layer of the network to convert the data into class categories (e.g., 0 and 1), the regression maps a set of inputs (contain continuous and discrete values) to the set of continuous outputs.

To perform the NN transformation, we carry out two

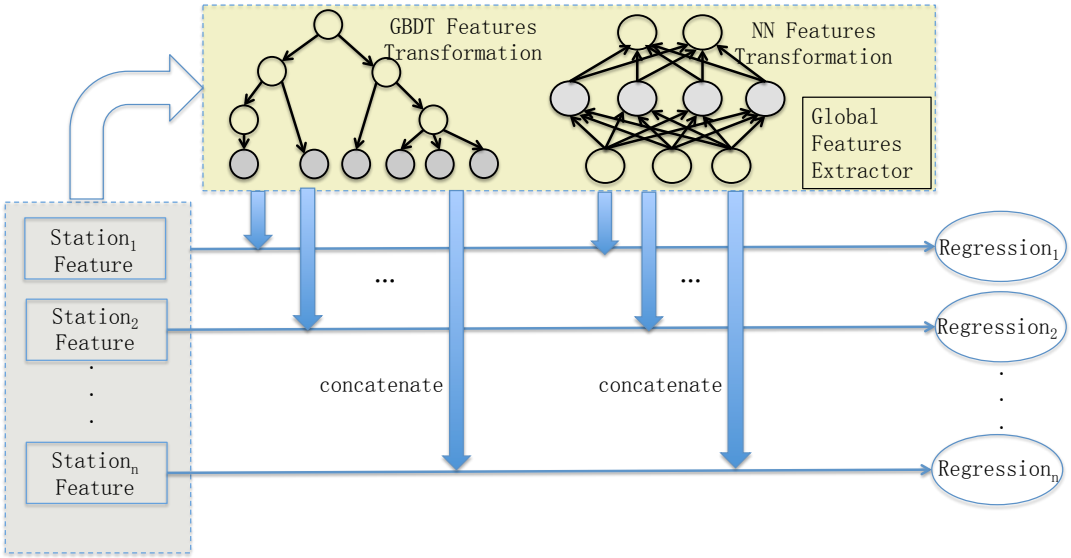


Figure 6: The station-centric model with global feature transformation framework. The decision tree and neural network are trained on the entire dataset. Then the global features (gray) are transformed by the models from the station-centric features and concatenated to the original features. The regression model will be trained for each station individually. In this way, we can integrate the station-centric features and global features.

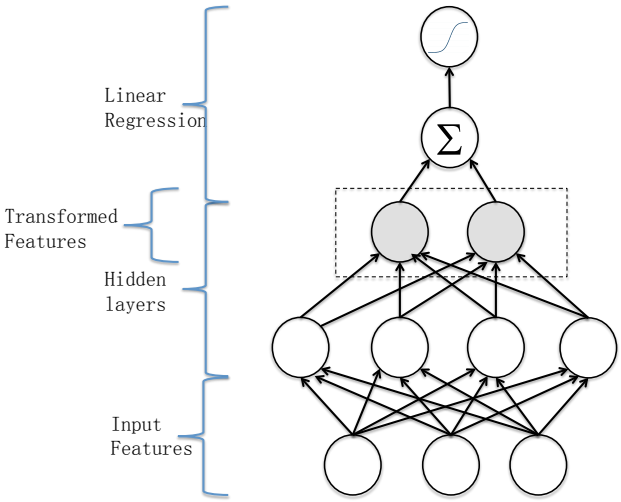


Figure 7: Input features are transformed by a two-layer (hidden) neural network. On top of the last hidden layer is a linear regression, which is used for fine-tuning the networks by back-propagation. The Nodes (gray) in the last layer of the network are concatenated to the original feature vector.

stages in the learning phase: pre-training and fine-tuning. We illustrate the two stages by considering the two-layer (two hidden layers) neural network in Figure 7. In the first stage, we use autoencoder to pre-train the model in an unsupervised manner. Each layer is trained by minimizing the error in reconstructing its input layer by layer. Once the first layer is trained, we can train the second by feeding the latent representation from the layer below. Once the pre-training is done in all layers, the network will go through a second learning stage called fine-tuning. We will use target values in the training data to fine-tune where we want to minimize prediction error on the supervised task. For this, we add a linear regression layer on top of the network. More precisely, it is the representation output from the second layer. Then we train the entire network by back-propagation. Different from the GBDT transformation, the NN transformation in this task converts a hybrid-valued (continuous and discrete values) vector into a real-valued vector. It also transforms the basic features to higher dimensional sparse features, also called over-complete representation [19, 29]. A sparse over-complete representations can be viewed as an alternative "compressed" representation to increase model's separability and interpretability [19, 17]. The output from last hidden layer is regarded as a new feature representation. The NN is also trained in a batch manner.

In the transformation phase, an instance is fed to the trained NN, then the output of first hidden layer will be the input of the second hidden layer. By the linear combination and non-linear mapping computation, the NN features are generated from the second hidden layer. The number of NN features equals to the number of nodes in this layer.

4.5 Summary

We summarize all the models and their purposes in Table 2. In summary, linear regression, support vector regression, decision tree and random forest are used in the regression task of demand prediction. Gradient boosting decision tree and neural network are used to extract global information, other algorithms such as principal component analysis (PCA) and linear discriminant analysis (LDA) can be used as feature extractors. Gradient boosting decision tree uses a hierarchical tree structure to encode that information from labeled data from all stations, while neural network uses multi-layer perceptron to incorporate the important patterns in labeled data from all stations.

Model	Function
Linear Regression	Demand Prediction
SVR	Demand Prediction
DT	Demand Prediction
RF	Demand Prediction
GBDT	Global Feature Extraction
NN	Global Feature Extraction

Table 2: A summary of all models and their functions as discussed in Section 4.

5. EXPERIMENTAL RESULTS

In this section, we validate our approach. We also study the performance of the city-centric model, the station-centric model and the station-centric model with global features when varying the amount of labeled training set. This will illustrate the limitations and advantages of different approaches.

5.1 Experimental Setting

For the comparisons of different approaches in Section 5.2, we used data from the entire 2015 to train a model so that it can incorporate the information of different seasons and factors. Then, we test it on the January and February data of 2016.

Dataset	# samples	# months	# stations
Training data	135,813	12	498
Test data	26,342	3	484

Table 3: A summary of the dataset used in Section 5.2.

For the global transformation models, we use GBDT with 1000 trees and obtain 9 additional features. For the neural network transformation, we use a two-layer fully connected network, $21-512-64$, the activation function in the nodes is "ReLU" function. We train the network by RMSprop gradient decent [26] with 100 iterations, the learning rate is 0.05 and the batch size is 32.

The regression model is trained in the original space [18]. We chose three metrics to evaluate our models, mean absolute error (MAE), root-mean-squared-error (RMSE) and root-mean-squared-log-error (RMSLE). [3] They are calculated as

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|p_i - y_i|,$$

$$\text{RMSE} = \sqrt{\frac{1}{n}(\sum_{i=1}^{n}(p_i - y_i))^2}$$

and

$$\text{RMSLE} = \sqrt{\frac{1}{n}(\sum_{i=1}^{n}(\log(p_i+1) - \log(y_i+1)))^2},$$

where n is the total number of observations in the test data set, p_i is the prediction, and y_i is the ground truth.

[3] 6 competitions including one about Bike Sharing Demand use RMSLE as evaluation criterion. See https://www.kaggle.com/wiki/RootMeanSquaredLogarithmicError

MAE gives us a number with which to determine by how many visits our predictions are off, so that we can interpret it easily. RMSE gives more penalties to the predictions which are far away form the true label. RMSLE penalizes under-prediction more than over-prediction. In an on-demand marketplace, it may be better to be safe than sorry, meaning it is better to over-prepare for demand peaks, than to not have enough resources to respond to high demand.

5.2 Comparisons of Approaches

In Table 4, the city-centric model, station-centric model and station-centric model with global features are compared. Four machine learning models are used in the evaluation and the MAE, RMSE and RMSLE are reported. For these results, we have several observations.

First, station-centric model with global features improves the prediction accuracy. Both GBDT and NN can provide useful improvement with global features, especially when evaluated by RMSE and RMSLE. Under RMSE, concatenating of GBDT and NN global features does not help too much compared to being evaluated by RMSLE. But when we use RMSLE in evaluation, global features by both GBDT and NN provide the best results. This means that a fusion of both GBDT and NN features can help improve accuracy in case of under-prediction. Under MAE, the improvements when using global features are less significant. This indicates that the global features are more useful in improving the city-centric and station-centric in the cases when the two models' predictions are far away from the ground truth.

Second, improvements can be observed for all four regression models, which suggests that our method works generally. Linear regression and random forest perform better than SVR and decision tree. The reason why SVR does not perform very well may be that we use the default parameter setting of SVR because SVR performance is known to be very sensitive to its parameter setting.

As linear regression and random forest provide the best performances, in the following experiments we focus on two regression models.

5.3 Varying Size of Training Dataset

In this section, we study the effects of varying the labeled training data size. Sometimes the labeled data for a particular station is very limited. For example, it is common to have newly built bike stations, as shown in Figure 4. When a station is newly built and little historical data is available, does a station-centic model still work well for this station? If not, could global features help improve prediction accuracy? To answer these questions, we design an experiment where we vary the amount of training data and study the achieved prediction accuracy.

Figure 8 shows the results. When we reduce the amount of training data, the performance of the city-centric model is relatively stable. However, the performance of the station-centric model degrades sharply. This indicates that in general, a station-centric model outperforms city-centric model. However, using the city-centric model can provide better predictions than using a station-centric model, when available training data is very limited. Making use of global features can compensate for the loss of accuracy to some extent, in the case when the labeled training data is not too sparse. For example, in (b) (c) (d) of Figure 8, when the percentage of data is around 15% (this corresponds to data collected

SVR

	City-Centric	Station-Centric	Station-Centric + GBDT	Station-Centric + NN	Station-Centric + GBDT + NN
MAE	68.5861	57.6646	**50.2394**	**50.2394**	**50.2394**
RMSE	83.3842	64.9557	58.3893	58.3893	**58.3883**
RMSLE	1.3965	0.8789	**0.7777**	**0.7777**	**0.7777**

Linear Regression

	City-Centric	Station-Centric	Station-Centric + GBDT	Station-Centric + NN	Station-Centric + GBDT + NN
MAE	26.5998	**17.3100**	**17.3100**	**17.3100**	**17.3100**
RMSE	40.1205	28.0785	23.4294	**24.1726**	**24.1726**
RMSLE	0.7559	0.6137	0.4870	**0.4718**	**0.4718**

Decision Tree

	City-Centric	Station-Centric	Station-Centric + GBDT	Station-Centric + NN	Station-Centric + GBDT + NN
MAE	31.6958	33.3640	33.6577	**30.7033**	33.2597
RMSE	51.4784	41.8557	41.8758	**40.1827**	40.2869
RMSLE	0.7952	0.7850	0.7845	0.7633	**0.7539**

Random Forest

	City-Centric	Station-Centric	Station-Centric + GBDT	Station-Centric + NN	Station-Centric + GBDT + NN
MAE	23.0282	19.5021	**19.4514**	19.4516	19.4992
RMSE	37.0090	28.2118	25.0652	**25.6240**	25.9390
RMSLE	0.6582	0.5412	0.4128	0.4048	**0.4045**

Table 4: A comparison between the city-centric model, the station-centric model, and the station-centic model with global features. The performance using four regression models is reported: support vector regression, linear regression, decision tree and random forest. GBDT and NN are used in the global features extraction.

over 2 months), the station-centric model with global features performs very well compared to the city-centric model.

5.4 Discussions

In this section, we discuss the approaches evaluated in this paper.

5.4.1 *When to use station-centric model with global feature transformation*

In general, the station-centric model with global feature transformation performs better than the city-centric and station-centric models. As Figure 8 indicates, for a particular bike station with limited labeled training data, the global approach provides the most stable performance. In the case when a new bike station is deployed and no bike usage information is available for it, the best practice is to train a model on the labeled data from all stations. Gradually, when more and more labeled data becomes available for this new bike station, we can switch to a station-centric model with global feature transform. In our empirical results, in the first two months when a new station is deployed, it is better to use a city-centric model to predict the demand. After about two months, we can use station-centric model with global feature transform instead.

5.4.2 *Comparison between GBDT and NN as global feature extractor*

The results in Table 4 suggest that GBDT and NN can extract useful features and improve the prediction performance. We report the correlation coefficients between each feature and the demand in Figure 9.

GBDT uses a hierarchical tree structure to encode information induced from labeled data from all stations, while NN uses multi-layer perceptrons to incorporate the important patterns. Taking advantage of the higher complexity of NN, features with higher coefficient than the original features are induced. As shown in Figure 9, four features extracted by NN have higher correlation coefficients, than the original handcraft features. The GBDT can also extract useful features from the labeled data from all stations. At the same time, GBDT is simpler than NN, thus the feature transform is more computationally efficient.

6. CONCLUSION AND FUTURE WORK

In this paper, we study the problem of demand predictions in a bike sharing system. This method is motivated by two observations. First, the user patterns of visits vary substantially between different stations. Second, it is common to have newly built bike stations with very limited labeled training data.

We present a method to improve the bike demand prediction in the NYC Citi Bike sharing system. This research could potentially assist service providers to optimize bike allocation and manage station maintenance. In our method, we adopt a station-centric model to optimize the prediction accuracy for particular bike stations. To overcome the limitation of a station-centric model when training data is sparse for a specific station, global features are extracted by GBDT or NN to enhance the accuracy and robustness of predictions. The empirical results indicate that our method can efficiently improve the performance in bike demand predictions.

Our global feature transform framework aims to solve the problem of different data patterns in different bike stations. In future, this framework of combining station-centric model

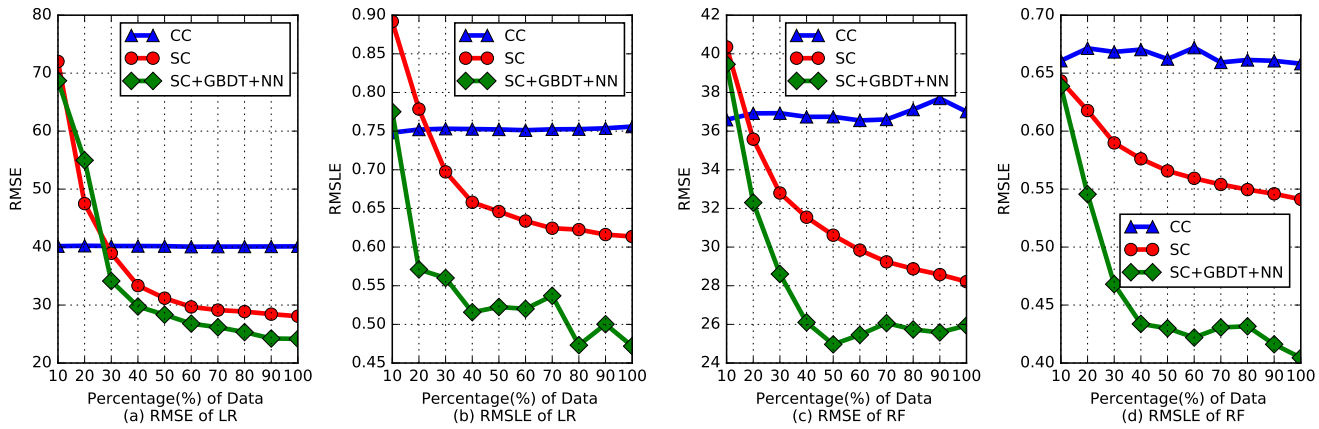


Figure 8: The amount of available labeled data's effects on performance for city-centric (CC), station-centric (SC) and hybrid (SC + GDBT + NN) models. The x-axis shows the percentage of training data when splitting the data by time. The y-axis shows the RMSE or RMSLE. The models of linear regression and random forest are used in the evaluations. We can observe that the CC model is robust to the amount of available labeled data, while SC model's performance degrades when we have less amount of labeled data. The model by SC + GDBT + NN is more robust in case when training data is limited.

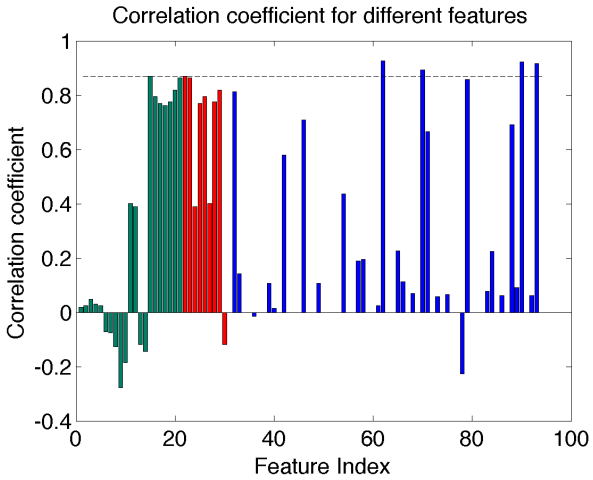


Figure 9: The correlation coefficients between features and demand. The features from Table 1 are to the left (green). The features extracted by GBDT and NN are in the middle (red) and to the right (blue) respectively. NN can extract some features with higher correlation coefficient, even compared to handcrafted features in Table 1.

and global features can also be explored to predict the number of check-in or check-out.

Predicting the visits would have practical applications. For example, the number of visits could tell us how frequently the bike docks and bikes are used in some stations. The maintenance of bike docks and bikes is the major cost of a bike sharing system. If we are going to more visits in some particular bike stations, we should pay more attention on those stations in repairing or replacement. A good understanding of those information can help optimize maintenance and reduce cost. So in future it could be interesting to have empirical studies to optimize the bike sharing system's maintenance based on the users' daily activity patterns.

Another future work using demand prediction could be that, for those stations going to have more daily visits, more bikes and more bike docks should be arranged in those stations. If the users find there are very limited bikes to borrow or limited bike docks to return bike, they tend to be unsatisfied about the bike sharing service. For those users using the bike sharing with fitness purpose, accurate predicting the amount of visits can guide us advertisement strategy in those stations. For example, if some stations are going to have more visits, it is more worth putting advertisement there and set higher price for the advertisement.

7. ACKNOWLEDGEMENT

This research is supported in part by the National Science Foundation under the award 1346066: SCH: INT: Collaborative Research: FITTLE+: Theory and Models for Smartphone Ecological Momentary Intervention.

8. REFERENCES

[1] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, and G. Penn. Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4277–4280. IEEE, 2012.

[2] Y. Bengio. Learning deep architectures for AI. *Foundations and trends in Machine Learning*, 2(1):1–127, 2009.

[3] A. L. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1):245–271, 1997.

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] H. Cheng and E. Cantú-Paz. Personalized click prediction in sponsored search. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 351–360. ACM, 2010.

[6] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio. Attention-based models for speech

recognition. In *Advances in Neural Information Processing Systems*, pages 577–585, 2015.

[7] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[8] A. Dhond, A. Gupta, and S. Vadhavkar. Data mining techniques for optimizing inventories for electronic commerce. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 480–486. ACM, 2000.

[9] A. Grover, A. Kapoor, and E. Horvitz. A deep hybrid model for weather forecasting. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 379–386. ACM, 2015.

[10] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, H. R. Atallah, Antoine, S. Bowers, and C. J. Quiñonero. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, pages 1–9. ACM, 2014.

[11] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[12] Z. Huang, G. Ren, and H. Liu. Optimizing bus frequencies under uncertain demand: Case study of the transit network in a developing city. *Mathematical problems in Engineering*, 2013.

[13] N. Kasabov. Global, local and personalised modeling and pattern discovery in bioinformatics: An integrated approach. *Pattern Recognition Letters*, 28(6):673–685, 2007.

[14] D. Kotzias, M. Denil, N. De Freitas, and P. Smyth. From group to individual labels using deep features. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 597–606. ACM, 2015.

[15] I. G. Kreft, I. Kreft, and J. de Leeuw. *Introducing multilevel modeling*. Sage, 1998.

[16] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10), 1995.

[17] M. S. Lewicki and T. J. Sejnowski. Learning overcomplete representations. *Neural computation*, 12(2):337–365, 2000.

[18] J. Liu, L. Sunu, W. Chen, and H. Xiong. Rebalancing bike sharing systems: A multi-source data smart optimization. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016.

[19] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision research*, 37(23):3311–3325, 1997.

[20] E. O'Mahony and D. B. Shmoys. Data analysis and optimization for (Citi) bike sharing. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[21] E. D. O'Mahony. *Smarter Tools For (Citi) Bike Sharing*. PhD thesis, Cornell University, 2015.

[22] S. W. Raudenbush and A. S. Bryk. *Hierarchical linear models: Applications and data analysis methods*, volume 1. Sage, 2002.

[23] T. Schneider. A tale of twenty-two million citi bike rides: Analyzing the nyc bike share system, 2016.

[24] S. Shankar, V. K. Garg, and R. Cipolla. Deep-carving: Discovering visual attributes by carving deep neural nets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3403–3412, 2015.

[25] D. Singhvi, S. Singhvi, P. I. Frazier, S. G. Henderson, E. O'Mahony, D. B. Shmoys, and D. B. Woodard. Predicting bike usage for New York Citys bike sharing system. In *AAAI 2015 Workshop on Computational Sustainability*, 2015.

[26] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop. *COURSERA: Neural networks for machine learning*, 2012.

[27] V. Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2013.

[28] G. M. C. Viglioni, M. V. Q. Cury, and P. A. L. da Silva. Methodology for railway demand forecasting using data mining.

[29] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010.

[30] H. Wang, X. He, M.-W. Chang, Y. Song, R. W. White, and W. Chu. Personalized ranking model adaptation for web search. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 323–332. ACM, 2013.

[31] K. Xu, J. Ba, R. Kiros, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *The Journal of Machine Learning Research*, 2015.

[32] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In *Mobile Computing, Applications and Services (MobiCASE), 2014 6th International Conference on*, pages 197–205. IEEE, 2014.

[33] C. Zhou, P. Dai, and R. Li. The passenger demand prediction model on bus networks. In *Data Mining Workshops (ICDMW), 2013 IEEE 13th International Conference on*, pages 1069–1076. IEEE, 2013.