Accident Analysis and Prevention 141 (2020) 105551

Contents lists available at ScienceDirect

Accident Analysis and Prevention

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

Improving the transferability of the crash prediction model using the TrAdaBoost.R2 algorithm

Dongjie Tang[a], Xiaohan Yang[b], Xuesong Wang[a,c,*]

[a] School of Transportation Engineering, Tongji University, 4800 Cao'an Road, Jiading District, Shanghai, 201804, China
[b] School of Mathematics Science, Tongji University, 1239 Siping Road, Yangpu District, Shanghai, 200092, China
[c] The Key Laboratory of Road and Traffic Engineering, Ministry of Education, China

ARTICLE INFO

Keywords:
Crash prediction model
Transferability
TrAdaBoost.R2
Calibration factor
Negative binomial model

ABSTRACT

The crash prediction model is a useful tool for traffic administrators to identify significant risk factors, estimate crash frequency, and screen hazardous locations, but some jurisdictions interested in traffic safety analysis can collect only limited or low-quality data. Existing crash prediction models can be transferred if calibrated, but the current aggregate calibration method limits prediction accuracy and the disaggregate method is resource-consuming. Transfer learning is another approach to calibration that acquires knowledge from old data domains to solve problems in new data domains. An instance-based transfer learning technique, TrAdaBoost.R2, is adopted in this study since it meets the requirement of site-based crash prediction model transfer. TrAdaBoost.R2 was compared with AdaBoost.R2 using a simply pooled data set to examine the efficiency in extracting knowledge from a spatially outdated source data domain (old data domain). The target data domain (new data domain) was sampled to test the technique's adaptability to small sample size. The calibration factor method based on a negative binomial model was employed to compare its predictive performance with that of the transfer learning technique. Mean square error was calculated to evaluate the prediction accuracy. Two cities in China, Shanghai and Guangzhou, were taken mutually as source data domain and target data domain. Results showed that the models constructed with TrAdaBoost.R2 had better prediction accuracy than the conventional calibration method. The TrAdaBoost.R2 is recommended due to its predictive performance and adaptability to small sample size. Crash prediction models are proposed to construct for peak and off-peak hours separately.

1. Introduction

Crash prediction models play an indispensable role in traffic safety management, including significant risk factor identification, crash frequency estimation, and hazardous location screening. In 2010, the Highway Safety Manual (HSM) (AASHTO, 2010) was published in the United States to help agencies make decisions about transportation design, operation, and maintenance. Using the negative binomial modeling structure, the HSM proposes a set of crash prediction models by roadway facility type and site type, which are also known as safety performance functions (SPFs). However, since the HSM crash prediction models are developed using data from several specific states in the United States, they are based on certain traffic characteristics, traffic environments, and driving behaviors. Other jurisdictions interested in traffic safety analysis are therefore not encouraged to apply the models directly, and so the issue of model transferability has arisen.

Model transferability is premised on the difficulty in collecting new data for developing jurisdiction-specific crash prediction models. The conditions of the actual application are that little or low-quality data are available in the target region, and it is resource-consuming to collect a complete data set. Studies and projects, therefore, hope to borrow data from other regions to conduct traffic safety analysis.

Under the negative binomial model framework, jurisdiction-specific crash prediction models are established. Calibration methods are adopted to localize the available models to specific conditions. The most widely used method, calibration factor, requires only crash count data, but this limits prediction accuracy. To calibrate the predicted crash count more accurately, efforts have been made to develop more efficient calibration methods, such as the empirical Bayes (EB) method (Farid et al., 2016), local regression method (Farid et al., 2018), pooled data (Farid et al., 2016), and the Bayesian regression model with informative priors (Farid et al., 2017). Though the aforementioned disaggregate calibration methods achieve progress in improving predictive performance, few methods consider their feasibility in data collection.

* Corresponding author.
E-mail addresses: 645094630@qq.com (D. Tang), xiaohyang@tongji.edu.cn (X. Yang), wangxs@tongji.edu.cn (X. Wang).

https://doi.org/10.1016/j.aap.2020.105551
Received 27 November 2019; Received in revised form 5 April 2020; Accepted 10 April 2020
Available online 23 April 2020

In other words, the application of these methods tends to require a large quantity of data from the target region, which usually makes the methods impracticable.

Transfer learning can be an alternative to the current calibration methods. The Broad Agency Announcement (BAA) of the Defense Advanced Research Projects Agency (DARPA) defined transfer learning as the ability to recognize and apply knowledge and skills extracted in previous tasks to novel tasks (Pan and Yang, 2009). The method is inspired by the fact that people can apply knowledge learned previously to innovatively solve a new problem. Because transfer learning can acquire knowledge efficiently from old data domains to solve problems in new data domains, this study proposes to examine its viability for crash prediction model transfer.

An instance-based transfer learning technique, TrAdaBoost.R2 (Dai et al., 2007), was adopted for the study since it meets the requirements of site-based crash prediction model transfer. To transfer a crash prediction model, the first problem is that the source data (old data) borrowed from some regions may be spatially or temporally outdated. TrAdaBoost.R2, however, allows the distributions of data from different domains to be heterogeneous and learns efficiently from the useful part of the source data. The second problem is that limited target data are available, so building a new model by resource-consuming data collection is not feasible. TrAdaBoost.R2 can combine a small quantity of target data with the knowledge extracted from the source data to construct a powerful model. As transfer learning is adaptative to data sets with different distributions and small sample sizes, it makes sense to employ transfer learning techniques for crash prediction model transfer.

Department of Transportation in Shanghai has established a good data foundation. Using this data, a series of SPFs for different road facilities have been developed (Xie et al., 2013; Wang et al., 2015; Li and Wang, 2017; Yu et al., 2017; Wang et al., 2018a, b, c). Nevertheless, most cities in China have not begun their traffic safety analyses due to the data collection dilemma. Based on cities such as Shanghai, we hope to build crash prediction models for regions lacking data.

The main objective of this paper is to examine the viability of the transfer learning technique for crash prediction model transfer. Two aspects were proposed in evaluating the performance of the transfer learning algorithm: prediction accuracy and sample size. TrAdaBoost.R2 was compared with AdaBoost.R2 using a simply pooled data to examine efficiency in extracting knowledge from a spatially outdated source data domain (old data domain). The target data domain (new data domain) was sampled to test the technique's adaptability to small sample size. The calibration factor method based on a negative binomial model was employed to compare its prediction accuracy with that of the TrAdaBoost.R2. Two cities in China, Shanghai and Guangzhou, were mutually used as source data domain and target data domain. Geometric design features, traffic characteristics, and crash data were collected for urban arterials in the two cities.

2. Literature review

2.1. Calibration methods based on regression models

The development of calibration factors is the most commonly used calibration method. The HSM defines a calibration factor as the ratio of total observed and predicted crashes. The rationale behind calibration factors is to assume a linear relationship between observed crashes and predicted crashes.

As an aggregate method, a calibration factor is convenient to implement due to the availability of crash count data in the United States. SPFs have been transferred to many states using the calibration factor approach, such as Arizona (Srinivasan et al., 2016), Utah (Brimley et al., 2012), Missouri (Sun et al., 2014), North Carolina (Srinivasan and Carter, 2011), and Oregon (Xie, 2011). In a more comprehensive study, Farid et al. established SPFs for Florida, Ohio, Illinois, Minnesota, California, Washington, and North Carolina, and assessed each state's transferability to other states (Farid et al., 2018). Meanwhile, international model transferability research has also been conducted. In Italy (Cafiso et al., 2012), Brazil (La Torre et al., 2019), Saudi Arabia (Kaaf and Abdel-Aty, 2015), and China (Li et al., 2018) the calibration factor was also used to localize the SPFs in HSM.

The calibration factor's assumption of a linear relationship between observed and predicted crashes may help reduce prediction errors in some instances while causing additional errors in others. Multiplying a fixed coefficient to all sites, which is an attribute of the aggregate method, constrains the predictive performance of the calibration factor.

To calibrate the prediction at a disaggregate level, methods such as the EB (Farid et al., 2016) and local regression method (Farid et al., 2018) have been developed. The EB method demands traffic volume and segment length data from the target domain for calculating predicted crashes. Afterward, observed crashes in the target domain are combined with predicted crashes to obtain the expected crashes. Using different crash types, Farid et al. (2016) established SPFs for rural divided multilane highway segments in Florida, Ohio, and California. They found that the EB method enhanced prediction accuracy as compared with the calibration factor. However, if the target domain's volume and segment length data are available, as is necessary for the EB, a crash prediction model may be independently built for the target domain, which makes model transfer between the two regions unnecessary. Though Farid et al. (2016) suggested substituting historical crashes for observed crashes, the practicality of the EB method remains limited.

The essence of the Bayesian negative binomial model is pooled data, but its prediction is more efficient than pooling data directly. Bayesian negative binomial models are built with non-informative priors. The posteriors, which are the specified regression coefficients for the target domain, are assigned as the informative priors in the procedure of building the Bayesian negative binomial model for the source domain. Farid et al. (2017) introduced Bayesian inference to study model transferability for rural divided multilane highway segments in Florida and California. Their results demonstrated that models with informative priors predicted more accurately than models with non-informative priors. Nevertheless, little work has been done to investigate the performance of informative priors on model transferability if a small sample is collected.

Generally, the data employed to calibrate the available model is crucial to the calibration method. Methods requiring only limited data may reduce the prediction accuracy while methods requiring much data may lower feasibility. This study proposed that the tradeoff between predictive performance and feasibility in data collection should be considered cautiously.

2.2. Calibration methods based on transfer learning technique

AdaBoost, short for adaptive boosting, is a machine learning approach of which the main task is to generate a strong classifier from a series of weak classifiers to boost prediction accuracy (Freund and Schapire, 1996). The AdaBoost mechanism trains a weak learner from the training sample with current weight. The weight of the training sample is updated according to the error rate so that a later learner focuses on the training sample that has a high error rate. The later weak learner is trained on the training set with adjusted weight. Finally, the weak learners are integrated to form the ultimate strong learner. Drucker (1997) compared boosting with a single learner and bagging on the Friedman functions, and concluded that boosting was superior to the other two methods in handling the regression problem.

TrAdaBoost extends AdaBoost to the field of transfer learning (Dai et al., 2007). It allows learning from source data and helps construct a new model for target data. Though the distributions of source data and target data are assumed to be different, the weight of source data is automatically leveraged. Source data with similar or the same

distribution as the target data will be given a higher weight, which results in greater confidence in the whole model. In Dai et al. (2007)' s research, different ratios between same-distribution and different-distribution training data were examined. The error rates of TrAdaBoost were consistently lower than those of other algorithms based on Support Vector Machine (SVM) across different data sets.

Since the original TrAdaBoost algorithm was designed to manage the classification issue, the TrAdaBoost framework was combined with AdaBoost.R2 to develop the TrAdaBoost.R2 to manage the regression problem. However, Pardoe and Stone (2010) found two problems when executing the TrAdaboost.R2 algorithm. First, the weights of source instances that were similar to the target data tended to be reduced to zero. Second, the outliers, or target instances most dissimilar to source data, were assigned with higher weight. Hence, Pardoe and Stone introduced a two-stage TrAdaBoost.R2 algorithm to solve the two problems. For the first problem, AdaBoost.R2 is executed to gradually reduce the weight of source instances until a certain step is reached, which is determined by cross-validation. For the second problem, AdaBoost.R2 is executed to adjust the weights of target instances while the weights of source data are frozen. In Pardoe and Stone's study, the two-stage TrAdaBoost.R2 was tested on four data sets. It outperformed ExpBoost.R2 and transfer stacking algorithms.

3. Data preparation

This study's data are composed of geometric design features, traffic characteristics, and crash data from selected urban arterials. Only arterials with four lanes or more were chosen. Elevated roads and tunnels were excluded due to their geometric design differences from common urban arterials.

Then, the full data set consists of 18 urban arterials in downtown Shanghai and 21 in downtown Guangzhou. These urban arterials were divided by consecutive signalized intersections. Each divided arterial was then split where the number of lanes, presence of medians, or presence of non-motorized lanes was different. The above procedures ensured the segments to be homogenous in terms of roadway geometric design features. The arterials in Shanghai and Guangzhou were finally split into 176 and 183 segments respectively.

The geometric design features collected included segment length, number of lanes, presence of medians, and presence of non-motorized lanes. Segment length was extracted from a GIS base map; the number of lanes, presence of medians, and presence of non-motorized lanes were obtained from Google Earth® and field surveys.

To include the changing influence of explanatory variables during different time periods, traffic volume and average speed were collected for peak hours (PH) (07:00 − 09:00) and off-peak hours (OPH) (12:00–14:00) for the year 2015. Traffic volume and average speed were obtained from loop detectors and floating car data in Shanghai; in Guangzhou, the data were obtained from loop, wave, and magneto detectors. Summary statistics for the obtained variables are shown in Table 1.

The crash counts for all the segments were acquired during the peak or off-peak hours in 2015. Intersection-related crashes were excluded. Crashes were then joined to the segments spatially by using the spatial join toolbox in ArcMap®.

The mean crash counts in Shanghai during peak hours and off-peak hours were 6.26 and 4.32, respectively, while they were 5.22 and 8.22 in Guangzhou. The variances in crash count in Shanghai during peak hours and off-peak hours were 98.86 and 29.39, respectively, while they were 61.57 and 135.76 in Guangzhou.

4. Methodology

4.1. AdaBoost.R2

AdaBoost promotes a weak learning algorithm to a strong learning algorithm. The improvement of the algorithm involves a forward stagewise additive modeling algorithm. The additive model means that the strong estimator is linearly added by a series of weak estimators, as shown in Eq. (1).

$$F_T(x) = \sum_{t=1}^{T} \beta_t G(x, a_t) \tag{1}$$

$G(x, a_t)$ is the weak estimator, where a_t is the optimal parameter learned by the weak estimator, and β_t is the weight of the weak estimator in the strong estimator.

The forward stagewise additive modeling algorithm is a greedy approach in that the estimator generated by the next iteration is trained based on the previous iteration, as shown in Eq. (2). It iteratively minimizes the exponential loss function $L(y_i, F_T(x))$ and fits the following equations where N is the sample size.

$$\{\beta_t, a_t\} = argmin \sum_{i=1}^{N} L(y_i, F_{t-1}(x) + \beta_t G(x, a_t)) \tag{2}$$

In the regression problem, we take AdaBoost.R2 (Drucker, 1997) as a representative algorithm. AdaBoost.R2 changes the weight of the training data, which is the probability distribution of the sample. The idea is to decrease the sample weights that are correctly classified and increase those that are misclassified.

To predict an instance, AdaBoost.R2 adopts the weighted majority voting method. The smaller the error rates of weak estimators, the more weight these estimators take in the voting procedure. This is reasonable since the weak estimator with a good score should have a stronger voice.

The main steps of AdaBoost.R2 are as follows:

a. Initialize the weight distribution of the training data:

$$D_1 = (w_{1,1}, ..., w_{1,i}, ..., w_{1,n}),\ w_{1,i} = 1/n,\ i = 1,2, ...,n \tag{3}$$

For t = 1, 2..., T:

b. Construct a regression estimator $G_t(x)$ from a training data set with weight distribution D_t.

c. Calculate the error rate $e_{t,i}$ of $G_t(x)$ on the training data set. Here, we use the linear error rate. Note that the error ranges from zero to one.

$$D_t = \max_{i=1}^{n} |y_i - G_t(x_i)| \tag{4}$$

$$e_{t,i} = |y_i - G_t(x_i)|/D_t \tag{5}$$

d. Calculate the average error ε_t of estimator $G_t(x)$. If $\varepsilon_t \geq 0.5$, stop and set T = t − 1.

$$\varepsilon_t = \sum_{i=1}^{N} e_{t,i} w_{t,i} \tag{6}$$

e. Calculate the weight β_t of estimator $G_t(x)$. β_t is a measure of confidence of the estimator. A lower β indicates higher confidence.

$$\beta_t = \varepsilon_t/(1 - \varepsilon_t) \tag{7}$$

f. Update the weight distribution of the training data set. Here, the factor z_t is a normalizing constant.

$$w_{t+1,i} = w_{t,i}\, \beta_t^{1-e_{t,i}}/\, z_t,\ i = 1,2, ..., n \tag{8}$$

g. Output:

$F_T(x)$ = the weighted median of $G_t(x)$ for $1 \leq t \leq T$, using $\ln(1/\beta_t)$ as the weight.

4.2. TrAdaBoost.R2

As an extension of AdaBoost.R2, TrAdaBoost.R2 is an instance-based transfer learning algorithm (Dai et al., 2007). The training data for TrAdaBoost is partitioned into two groups, T_{source} $(i = 1, ..., n)$ and T_{target} $(i = n + 1, ..., n + m)$. T_{target} is assumed to reflect the same-distribution of the test data domain. T_{source} is an unknown-distribution data domain, which indicates that some data may be useful in learning knowledge while some data may be harmful. Considering the limited

Table 1
Summary statistics for obtained variables.

Variable	Description	Summary statistics	
		Shanghai	Guangzhou
Geometric design features			
Segment length	Length of segment (km)	mean: 0.384 SD: 0.269	mean: 0.611 SD: 0.282
Number of lanes	Lanes counted in both directions	4/5 lanes: 64 (36 %) 6/7 lanes: 39 (22 %) 8/9 lanes: 66 (38 %) 10 lanes: 7 (4%)	4/5 lanes: 42 (23 %) 6/7 lanes: 73 (40 %) 8/9 lanes: 57 (31 %) 10 lanes: 11 (6%)
Presence of medians	1 with median 0 without median	1: 134 (76 %) 0: 42 (24 %)	1: 166 (91 %) 0: 17 (9%)
Presence of non-motorized lanes	1 with non-motorized lane 0 without non-motorized lanes	1: 127 (72 %) 0: 49 (28 %)	1: 78 (43 %) 0: 105 (57 %)
Traffic characteristics			
Average volume PH	Average traffic volume per lane (pcu/h/lane)	mean: 448.88 SD: 108.48	mean: 378.09 SD: 224.94
Average volume OPH	Average traffic volume per lane (pcu/h/lane)	mean: 361.22 SD: 94.68	mean: 357.59 SD: 225.62
Average speed PH	Mean speed on the segment during peak hours (km/h)	mean: 26.89 SD: 10.05	mean: 28.81 SD: 32.17
Average speed OPH	Mean speed on the segment during off-peak hours (km/h)	mean: 31.70 SD: 10.09	mean: 33.92 SD: 32.34

Note: SD is the standard deviation. Min is Minimum. Max is Maximum.

sample size of T_{target}, it is supplemented with T_{source} to help build the model. The main difference between AdaBoost and TrAdaBoost is the strategy by which they update the sample weights of training data. For T_{target}, both algorithms reduce the weights of high-error-rate instances and increase the weights of low-error-rate instances. For T_{source}, however, TrAdaBoost.R2 reduces the weights of high-error-rate instances since they may conflict with test data and the weights of low-error-rate instances will be increased to contribute more toward learning.

The main steps of TrAdaBoost.R2 are as follows:

a. Initialize the weight distribution of the training data:

$$D_1 = (w_{1,1},...,w_{1,n+i},...,w_{1,n+m}),\ w_{1,n+i} = \frac{1}{n+m},\ i = 1,2,...,n+m \tag{9}$$

For t = 1, 2..., T:

b. Construct a regression estimator $G_t(x)$ from a training data set with weight distribution D_t.

c. The average error is only calculated on T_{target}. Here n is the size of source data and m is the size of the target data. If $\varepsilon_t \geq 0.5$, stop and set T = t − 1.

$$\varepsilon_t = \sum_{i=n+1}^{n+m} \frac{w_{t,i}\ |y_i - G_t(x_i)|}{\sum_{i=n+1}^{n+m} w_{t,i}} \tag{10}$$

d. Calculate β_t for T_{target} and T_{source} respectively.

$$\beta_t = \begin{cases} 1/(1+\sqrt{2\ln n/T},\ for\ T_{source} \\ \varepsilon_t/(1-\varepsilon_t),\ for\ T_{target} \end{cases} \tag{11}$$

e. Update the weight distribution of the training data set.

$$w_{t+1,i} = \begin{cases} w_{t,i}\beta_t^{|y_i - G_t(x_i)|},\ for\ T_{source} \\ w_{t,i}\beta_t^{-|y_i - G_t(x_i)|},\ for\ T_{target} \end{cases} \tag{12}$$

f. Output:

$F_T(x)$ = the weighted median of $G_t(x)$ for $[T/2] \leq t \leq T$, using $\ln(1/\beta_t)$ as the weight.

4.3. Two-stage TrAdaBoost.R2

Recall the two problems found by Pardoe and Stone (Pardoe and Stone, 2010) when executing the TrAdaboost.R2 algorithm: the weights of source instances similar to the target data tended to be reduced to zero, and the outliers were assigned with higher weight. To solve these two problems of extreme weight distribution in the training data set, they introduced a two-stage TrAdaBoost.R2 algorithm. In the first stage, the total weight of T_{source} is adjusted by steps. This stepped reduction of total source weight is designed to approach zero when the last step, S, is reached. β_k is then determined to satisfy the above condition by a binary search instead of a calculation based on error rate. Then, the weight distribution of T_{source} is updated according to error rates. In the second stage, the weight distribution of T_{target} is updated by executing the common AdaBoost.R2 while the weight of T_{source} remains unchanged.

The main of the two-stage TrAdaBoost.R2 are as follows:

a. Initialize the weight distribution of the training data:

$$D_1 = (w_{1,1},...,w_{1,i},...,w_{1,n+m}),\ w_{1,i} = \frac{1}{n+m},\ i = 1,2,...,n+\mathrm{m} \tag{13}$$

For k = 1, 2..., S:

b. Construct $Model_k$:

Call AdaBoost.R2 with $T = T_{source} + T_{target}$, a base regression estimator $G(x)$, and the weight vector w_k. In this procedure, T_{source} stays unchanged. Calculate the $error_k$ of $Model_k$ by F-fold cross-validation.

c. Call a base regression estimator $G(x)$ with T and weight vector w_k.

d. Calculate error rate $e_{k,i}$ for each instance in T.

e. Update the weight distribution as follows:

$$w_{k+1,i} = \begin{cases} w_{k,i}\beta_k^{e_{k,i}}/Z_k,\ forT_{source} \\ w_{k,i}/Z_k,\ forT_{t\,\mathrm{arg}\,et} \end{cases} \tag{14}$$

Here, Z_k is a normalizing constant, and β_k is designed such that the total weight of T_{target} is $\frac{m}{n+m} + \frac{k}{(S-1)}(1 - \frac{m}{(n+m)})$. It should be noted that β_k is larger than zero and smaller than one. This indicates that the instance with a small error rate $e_{k,i}$ will have a larger updated weight value, which is consistent with the concept of TrAdaBoost.R2.

Model Output:

$F(x) = F^k(x)$ where k= argmin $\mathrm{error_k}$

4.4. Negative binomial model

Crash counts are assumed to follow the negative binomial (NB) distribution. The use of the NB model rather than the Poisson model is

to handle the over-dispersion of the crash observations. Under the NB model, the probability of observing y_i crashes at segment i is defined as Eq. (15) (Lord and Miranda-Moreno, 2008):

$$p(Y_i = y_i) = \frac{\Gamma(y_i + 1/k_i)}{y_i \times \Gamma(1/k_i)} \times \left(\frac{1/k_i}{1/k_i + \mu_i}\right)^{1/k_i} \times \left(\frac{\mu_i}{1/k_i + \mu_i}\right)^{y_i} \quad (15)$$

where μ_i and k_i are expected crash counts and over-dispersion for segment i. $\Gamma(.)$ is the gamma function. When k_i approaches zero, the NB model converges to the Poisson model.

The function of expected crash counts and explanatory variables segment is described as Eq. (16):

$$\mu_i = exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ...++\beta_n X_n + \varepsilon) \quad (16)$$

where β_0 is the intercept. β_1, β_2...,β_n are the coefficients for each explanatory variable, ε is the random error term, and the exponential of ε follows gamma distribution as shown in Eq. (17):

$$\exp(\varepsilon) \sim \Gamma(1, k) \quad (17)$$

Considering that segment length may have different influence on parameter estimation (AASHTO, 2010; Farid et al., 2016), over-dispersion is defined as a function of segment length, L_i. The functional form is shown in Eq. (4). φ is a regression coefficient as Eq. (18).

$$k_i = \frac{1}{\varphi \times L_i} \quad (18)$$

The calibration factor (CF) is the ratio of the total number of observed crashes N_{obse} and predicted crashes N_{pred}, as shown in Eq. (19):

$$CF = \sum N_{obse} / \sum N_{pred} \quad (19)$$

4.5. Prediction accuracy

The method chosen to evaluate prediction accuracy was a mean square error (MSE), which is defined as Eq. (20):

$$MSE = \sum_{i=1}^{n} (N_{observed} - N_{predicted})^2 / n \quad (20)$$

where $N_{observed}$ is the observed crash count and $N_{predicted}$ is the crash count predicted by the model.

5. Model comparison

The goal of model comparison was to examine the performance of the transfer learning technique from two aspects, prediction accuracy and adaptability to small sample size. To determine the temporal and spatial features in model transfer, models were constructed from the perspective of two dimensions separately, i.e. model transfer during peak hours and off-peak hours, and model transfer between the two cities in both directions.

In a given case, one city was assumed to be interested in traffic safety analyses, but insufficient data were available, and the other city had undertaken studies on crash prediction and collected sufficient data. Thus, when Guangzhou was taken as the first city above, it was the target data domain; Shanghai, as the second city, was the source data domain. Shanghai data was then added to the Guangzhou data. Using the mixed data, the model was trained to predict crashes in the target domain. It should be noted that the positions of the two cities were changed in the other case.

HSM defines the desirable minimum sample size for the calibration data set is 30–50 sites, which indicates a reasonable sample size to collect in the target domain. Under this circumstance, 30 % of the target domain data were taken (about 50 segments in this study) as a training data set since a higher percentage consumes substantial data resources. Because source domain data was always available in the assumption, the training data set was supplemented with 100 % of the source domain's data to build Model 1 using two-stage TrAdaBoost.R2.

Note that AdaBoost.R2 was employed in Model 2 with the same training data. Since AdaBoost.R2 is not a transfer learning algorithm, the essential way Model 2 analyzed the target data and the source data was simply pooled data. The comparison between AdaBoost.R2 and two-stage TrAdaBoost.R2 helped to investigate how well the two-stage TrAdaBoost.R2 acquired knowledge from the source data.

Other models were also developed for comparison. AdaBoost.R2 was used for local (target domain data only) Models 3 and 4 as well. Model 3, with 70 % of the target domain data, was expected to show better prediction accuracy due to a larger sample from the target domain. Model 4, with 30 % of the target domain data, was constructed to test whether the source data enhanced the performance of the models. Negative binomial models were also employed to compare them with AdaBoost.R2 and two-stage TrAdaBoost.R2. Model 6 was a local model, using 100 % of data from the source domain only. The same source data was used for Model 5, along with crash count data from the target domain. To employ the common calibration method in HSM, the calibration factor was calculated from the crash data to improve the model's transferability.

Note that the functions of Model 1, Model 2, and Model 5 are transfer models; and the functions of Model 3, Model 4, and Model 6 are local models. The models were constructed using Python (Van Rossum and Drake, 2011), and are summarized in Table 2.

To evaluate prediction accuracy, MSE was calculated as the average of 10 random repeated selection of subsets of training and test data.

6. Results

6.1. Parameter training

Two classical algorithms were selected as the base estimators for AdaBoost.R2 and two-stage TrAdaBoost.R2: Decision Tree (DT) and Support Vector Machine (SVM).

The key parameter for DT is the max-depth of the decision tree, which is defined as the number of features, 6 in our case.

The key parameters for SVM are the penalty parameter of the error

Table 2
Data domain and method settings for the six models.

Model	Training data	Test data	Method
Model 1 (transfer)	Source domain (100 %) Target domain (30 %)	Target domain (100 %)	Two-stage TrAdaBoost.R2
Model 2 (transfer)	Source domain (100 %) Target domain (30 %)	Target domain (100 %)	AdaBoost.R2
Model 3 (local)	Target domain (70 %)	Target domain (100 %)	AdaBoost.R2
Model 4 (local)	Target domain (30 %)	Target domain (100 %)	AdaBoost.R2
Model 5 (transfer)	Source domain (100 %) Crash count in Target domain (100 %)	Target domain (100 %)	Calibration factor based on NB model
Model 6 (local)	Source domain (100 %)	Target domain (100 %)	NB model

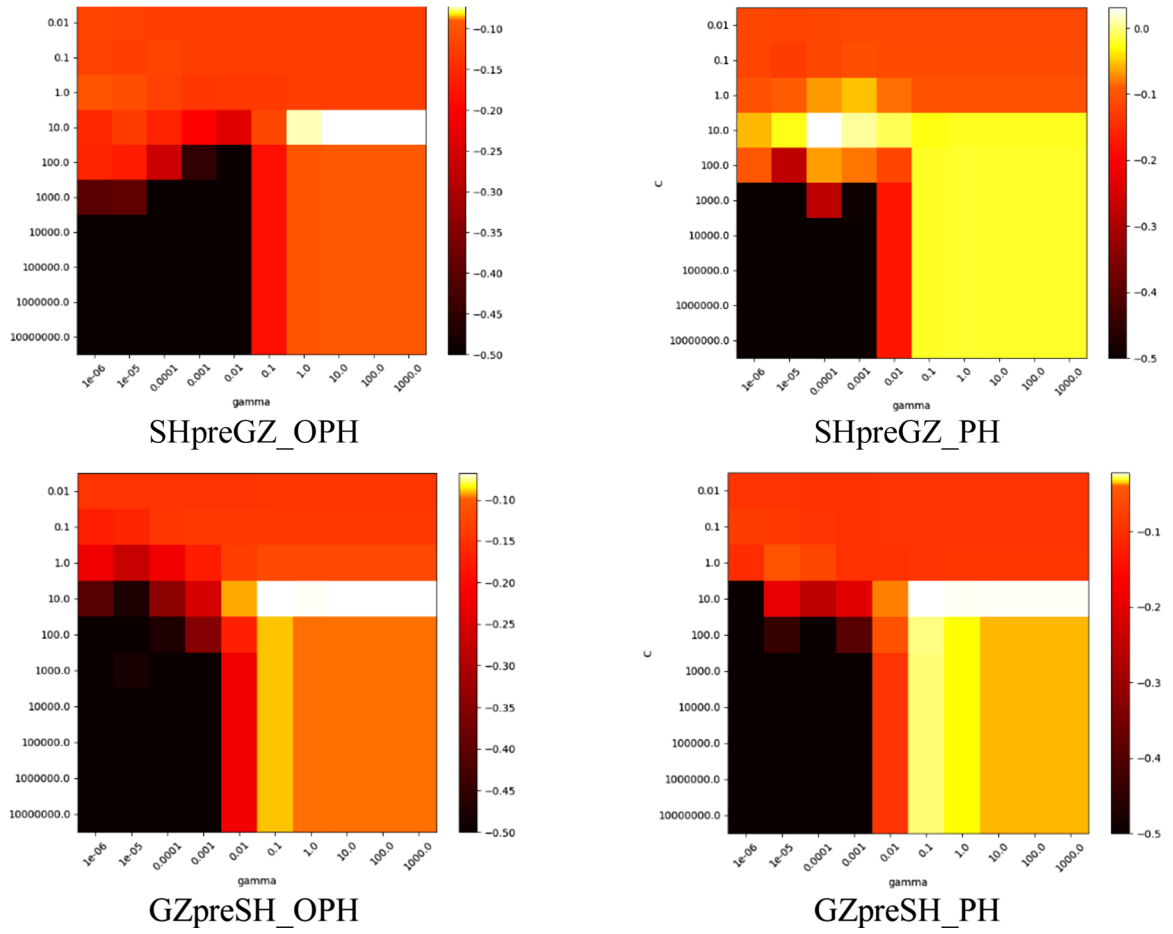

Fig. 1. Cross-validated grid-search results for SVM.

term, C, and the kernel coefficient for the radial basis function, gamma. A cross-validated grid-search method was used in this study to find the optimal parameters. Each cube is represented by a coordinate, which is C and gamma, as depicted in Fig. 1. A higher score of each pair of parameters (C and gamma) is visually represented by the lighter color. Note that the higher the score, the better the parameters. Thus, the optimal parameters can be chosen by selecting the cube with the lightest color.

The four cases in Fig. 1 denote the model's transfer direction and time. For example, when Guangzhou data are taken as target data, Shanghai data will be source data and be added into Guangzhou data. If this case is in the peak hours, we record it as SHpreGZ_PH. The same definition is applied to the other three cases. In SHpreGZ_PH case, the lightest cube was white where C and gamma were 10 and 10^{-4}. In another word, the optimal parameters of C and gamma were 10 and 10^{-4}.

AdaBoost has been demonstrated to be resistant to overfitting based on the *margins explanation* theory (Schapire, 2013). This powerful property ensures that we can choose a certain number of estimators for AdaBoost if the prediction accuracy on the test data converges.

The key parameter for the two-stage TrAdaBoost.R2 is the number of estimators, which influences the model's fitness and generalization capability. Thus, the performance of the number of estimators was tested, as shown in Fig. 2.

Though the curves are not quite smooth, they converge well in most cases. Fig. 2 shows that the value of MSE for DT generally decreased as the number of estimators increased. For DT, the MSE value dropped quickly by 10 estimators. When the number of estimators became greater than 20, the MSE value remained stable, which indicates that 20 estimators were sufficient for the algorithm based on DT. For the SVM algorithm, the nearly horizontal curves for SHpreGZ_OPH, SHpreGZ_PH, and GZpreSH_OPH implied that the prediction accuracy kept unchanged with the increase of estimators. This coincided with Chan et al. (2001)'s study that the boosting of SVM stopped when the base estimators were more than one. Nevertheless, as a number of estimators were larger than 30, the MSE value for GZpreSH_PH began to show an upward trend, which seemed to be an overfitting behavior. Hence, we think ten to twenty is the proper number of estimators.

6.2. Comparison results

6.2.1. Decision tree and support vector machine

After adjusting the parameters, AdaBoost.R2 and two-stage TrAdaBoost.R2 were conducted using the DT and SVM algorithms. The predictive performance for the two-stage TrAdaBoost.R2 and the three AdaBoost.R2 models (shown in Table 2) are presented in Table 3. For example, in SHpreGZ_PH case, the MSE value of DT-based Model 1 was 16.106 while that of Model 2 was 19.150, which indicates Model 1 was better than Model 2.

For both algorithms, Model 3 (AdaBoost.R2 with 70 % local target data) shows expected best accuracy since it was constructed with a high percentage of target data. In most cases (all but SVM-based GZpreSH_OPH), the MSE values of transfer Model 2 (AdaBoost.R2) are smaller than those of local Model 4, with an average difference of 1.57. This indicates that source data indeed helped predict crash counts for the target data domain, possibly due to the similarity of some target instances. Further, the MSE values of transfer Model 1 (two-stage TrAdaBoost.R2) are consistently smaller than those of transfer Model 2 (AdaBoost.R2), with an average difference of 2.97. This outcome is reasonable since the two-stage TrAdaBoost.R2 helps identify the source

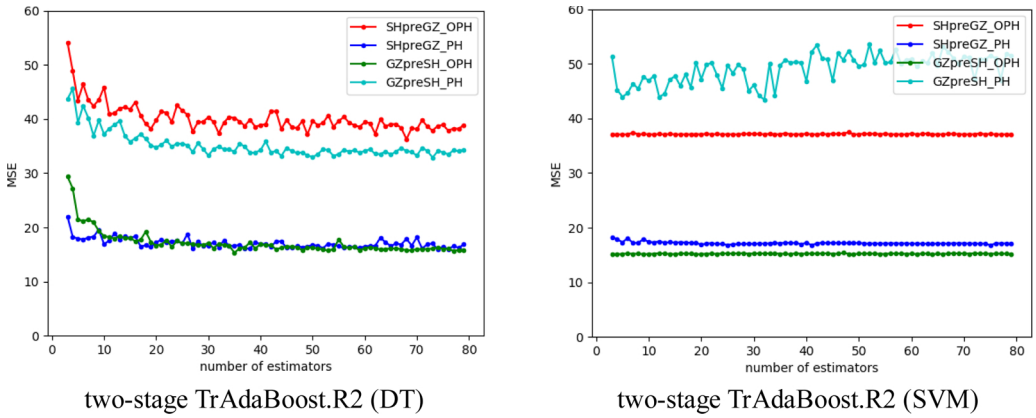

Fig. 2. Number of estimators for two-stage TrAdaBoost.R2.

Table 3
Prediction accuracy (MSE) of AdaBoost.R2 and two-stage TrAdaBoost.R2.

	Cases	SHpreGZ_OPH	SHpreGZ_PH	GZpreSH_OPH	GZpreSH_PH
DT	Model 1	36.330	16.106	15.803	32.975
	Model 2	40.207	19.150	18.570	35.427
	Model 3	25.618	12.582	10.898	23.045
	Model 4	40.208	21.752	21.269	37.636
SVM	Model 1	37.164	16.881	15.114	43.929
	Model 2	37.687	18.662	23.499	44.889
	Model 3	24.892	16.228	11.496	31.955
	Model 4	46.068	20.318	15.849	47.559

instances with the same-distribution and extracts more knowledge from source data than does AdaBoost.R2.

The models show similar trends for DT and SVM. The MSE values for the DT-based Model 1 are, in most cases, close to those for the SVM-based Model 1. Similar results can be found for Models 2, 3, and 4. Consequently, the relative performance of the two estimators is also similar.

Note that both DT-based and SVM-based models perform poorly in SHpreGZ_OPH, while they perform well in SHpreGZ_PH; and both model types also perform well in GZpreSH_OPH, while they perform poorly in GZpreSH_PH. These differences indicate the influence of time periods.

Overall, however, the prediction accuracy for both AdaBoost.R2 and two-stage TrAdaBoost.R2 models based on DT is better than those based on SVM, especially in the GZpreSH_PH case. Models based on DT were therefore chosen as the base for the following analysis.

6.2.2. AdaBoost.R2 and two-stage TrAdaBoost.R2

The error distributions of the four models based on the DT algorithm are illustrated in Fig. 3. The error is the difference between crash count and crash prediction. When the error is smaller than zero, the model overpredicts the crash count, and when the error is larger than zero, it underpredicts the crash count. Generally, when the crash count is smaller than seven or eight, the four models tend to overpredict the crash count, and when the crash count is larger than eight, the four models tend to underpredict the crash count. These errors contribute the most to the total error.

Fig. 3 shows that the models are relatively accurate in predicting instances when the crash count is smaller than about 15, while they produce higher errors when predicting instances with more than 15 crashes. Model 3 predicts the most instances of large crash counts correctly, which is reasonable since Model 3 is constructed with a high percentage of the target data set. The overall error distribution of Model 1, two-stage TrAdaBoost.R2, is similar to that of Model 3, and more accurately predicts instances with large crash counts than Models 2 and 4. Thus, Model 1 is demonstrably preferable among the transfer models.

6.2.3. Transfer learning technique and regression model in the four transfer cases

In addition to the transfer learning Models 1–4, NB crash prediction models, Models 5 and 6, were also constructed for the four different transfer cases (SHpreGZ_OPH, etc.). The calibration factor was applied to Models 5 to localize them. Fig. 4 summarizes the performance of all six models by case.

The calibration factor in Model 5 reduced the MSE values of Model 6 in all cases but SHpreGZ_OPH, for which the MSE value unexpectedly increases a bit after calibration. A study by Farid et al. (2016) showed similar results, suggesting that a larger error may be activated by multiplying a fixed factor to all instances. Still, both regression models have higher MSE values than the transfer learning Models 1–4. That is, the four models based on Adaboost.R2 and two-stage TrAdaboost.R2 predict crash for target data more accurately than the conventional regression models. Except for GZpreSH_OPH, both Adaboost.R2 and two-stage TrAdaboost.R2 improve the crash prediction accuracy substantially.

The transfer Model 2, with the assistance of 100 % of the source data, generally showed prediction error in all four of the cases lower than local Model 4, without the source data. While the local Model 3 had the lowest MSE values, outperforming the other models, it is resource-consuming to collect 70 % of target data, which reduces its feasibility. In comparison, the transfer Model 1 (two-stage TrAdaBoost.R2) requires just 30 % of target data and a source data set which is already available; and its MSE value is smaller than that of any other model except Model 3. It can be concluded that Model 1 performs well in predictive performance and adaptability to the small sample size.

An asymmetrical phenomenon was observed in peak and off-peak time periods for the Models 1–4 based on Adaboost.R2 and two-Stage Tradaboost.R2. If the prediction error was large when we took one data set as the source domain to predict crashes for the target domain data set, the prediction error would be relatively small when the roles of two data sets were reversed. As can be seen in Fig. 4, for example, the prediction error was large in SHpreGZ_OPH while the prediction error was relatively small in GZpreSH_OPH. This indicates that Guangzhou data are more transferable to Shanghai data in an off-peak time period. However, we can also observe that Guangzhou data are less transferable to Shanghai data in a peak time period in Fig. 3. Besides, we can also observe the same phenomenon from the MSE values of the SVM-based models in Table 3. This discrepancy in crash prediction accuracy indicates the necessity of a modeling strategy that we should consider different time periods if relevant data are available.

7. Discussion

AdaBoost has been derived by previous studies in a statistical view

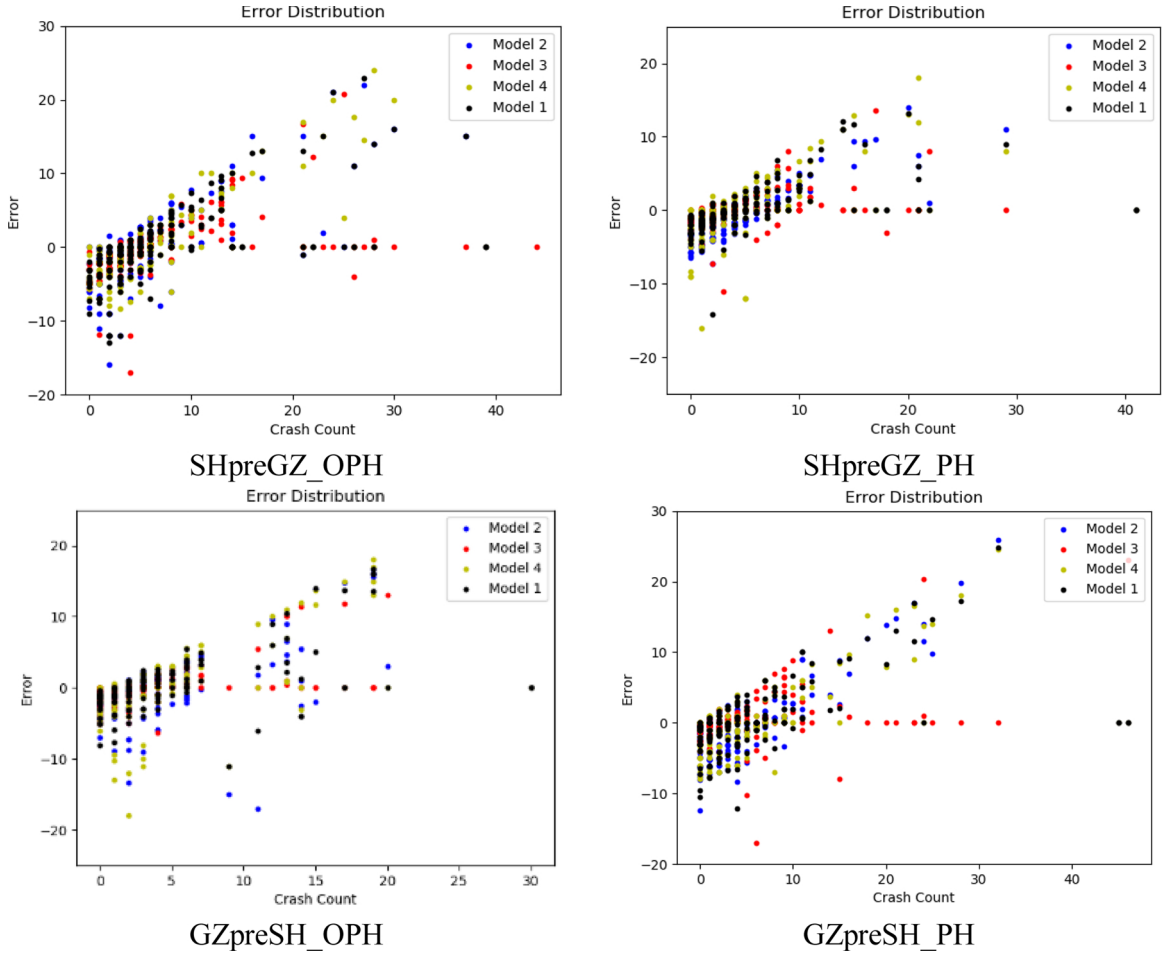

Fig. 3. Error distribution of models based on decision tree.

(Friedman et al., 2000; Zhou, 2014), one of which is based on statistical principles, namely additive model and maximum likelihood (Friedman et al., 2000). Based on the theory of AdaBoost, Dai et al. (2007) derived that TrAdaBoost iteratively reduces the prediction error on target data and the weighted training loss on source data gradually converges to zero.

TrAdaBoost can be broken down into three steps. First, a base estimator is chosen to fit the data. Similar to selecting the NB model to construct SPF, DT can be chosen as the base estimator for crash data. Second, the weights of the samples are adjusted. To minimize the exponential loss function, reassigning the weight of samples and focusing on high-error rate samples help to improve the predictive performance. Friedman et al. (2000) have proved that this step of updating the distribution of weight is an identical form of Newton-like update. A new base estimator will fit the crash data with reassigned weights. Third, the TrAdaBoost model is constructed with a set of base estimators. Though the prediction accuracy of a single DT may be worse than the NB model, TrAdaBoost combines several DTs to improve the accuracy of crash prediction.

As depicted in Table 3, the prediction accuracy for both AdaBoost.R2 and two-stage TrAdaBoost.R2 models based on DT is better than those based on SVM. Breiman (1996) studied on bagging algorithm, another ensemble method, and supplied reference on the superiority of an unstable learner (DT) over a stable learner (SVM). Such a

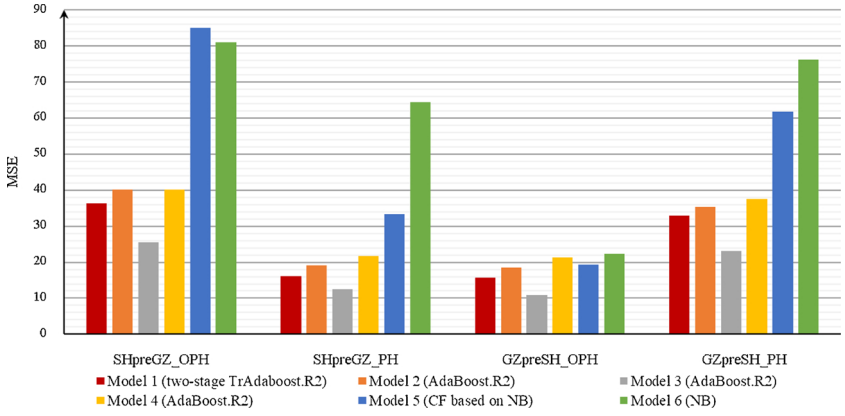

Fig. 4. Comparison of AdaBoost.R2, Two-Stage TrAdaboost.R2, and negative binomial models.

Table 4
Modeling results for guangzhou during peak and off-peak hours.

Parameter	Peak hours		Off-peak hours	
	Estimate	Pr > ChiSq	Estimate	Pr > ChiSq
Intercept	−2.42	< 0.001	−2.61	< 0.001
Segment length	0.62	< 0.001	0.53	< 0.001
Total volume in log form	0.60	< 0.001	0.46	< 0.001
Average speed	–	–	−0.01	0.003
Number of lanes [4–7] vs [8–11]	–	–	–	–
Presence of median Yes vs. No	−1.23	< 0.001	−1.21	< 0.001
Presence of non-motorized lanes Yes vs. No	–	–	–	–
φ	1.20	< 0.001	1.26	< 0.001

Note: –represents the removal of statistically insignificant variables at 95 % confidence interval.

conclusion can also be achieved for boosting algorithms. The experimental results from Bauer and Kohavi (1999) show that AdaBoost based on the unstable learner (DT) made a larger error reduction than the stable learner (Naïve-Bayes). Another research on boosting (Chan et al., 2001) also revealed that the accuracy of DT was higher than that of SVM, which is consistent with conclusions from Table 3.

Despite TrAdaBoost's adaptability to source data with heterogeneous distribution, the issue of transferability usually arises in different contexts considering NB models, which is also applied in this study. The reason Shanghai and Guangzhou could be considered as different contexts were therefore discussed.

The coefficient values of SPFs for both cities in Tables 4 and 5 reveal the different relationships between the crash and explanatory variables considering Shanghai and Guangzhou. As illustrated in Tables 4 or 5, the coefficient values over different time periods are similar for the same city. For example, the coefficients of traffic volume during peak and off-peak hours in Guangzhou were 0.60 and 0.46, respectively, which reveal a similar crash pattern in that city. However, the coefficients were relatively different between Guangzhou and Shanghai during the same time periods, which show that the two cities may have different crash patterns. For example, the traffic volume coefficient during peak hours in Guangzhou was 0.60, while in Shanghai it was 1.29. Note the similar comparisons for segment length.

The different contexts of Shanghai and Guangzhou can be referred to the significant discrepancy of coefficient values. On the other hand, this heterogeneity helps to prove the adaptability of TrAdaBoost to heterogeneous data since TrAdaBoost can extract useful information from the source domain and construct a stronger model than AdaBoost.

Table 5
Modeling results for shanghai during peak and off-peak hours.

Parameter	Peak hours		Off-peak hours	
	Estimate	Pr > ChiSq	Estimate	Pr > ChiSq
Intercept	−8.33	< 0.001	−7.27	< 0.001
Segment length	1.72	< 0.001	1.22	< 0.001
Total volume in log form	1.19	< 0.001	1.17	< 0.001
Average speed	–	–	−0.02	0.008
Number of lanes [4–7] vs [8–11]	−0.98	0.002	−0.64	0.031
Presence of median Yes vs. No	–	–	–	–
Presence of non-motorized lanes Yes vs. No	–	–	–	–
φ	1.22	< 0.001	1.47	< 0.001

Note: –represents the removal of statistically insignificant variables at 95 % confidence interval.

8. Conclusion

The transfer learning techniques of two-stage TrAdaBoost.R2 were evaluated in this study for prediction accuracy and adaptivity to small sample size. Four models were constructed using data from two cities in China, Shanghai and Guangzhou:

- Model 1 was constructed with two-stage TrAdaBoost.R2 using 100 % source data and 30 % target data;
- Model 2 was constructed with AdaBoost.R2 using 100 % source data and 30 % target data;
- Model 3 was constructed with AdaBoost.R2 using 70 % target data; and
- Model 4 was constructed with AdaBoost.R2 using 30 % target data.

Two negative binomial models, Model 5 and Model 6, were constructed with and without the calibration factor to compare conventional regression with the transfer learning models. All six models were constructed for peak hours and off-peak hours and between the two cities in both directions of the transfer.

The base estimators for two-stage TrAdaBoost.R2, DT and SVM, were investigated. Overall, the DT-based models were more accurate by a narrow margin and thus were preferred in this study.

The better performance of Model 2 (AdaBoost.R2 with 30 % target data and 100 % source data) than Model 4 (AdaBoost.R2 with 30 % target data only) reveals that the addition of source data can help predict crash counts for target cities. Further, better performance of Model 1 (two-stage TrAdaBoost.R2 with 30 % target data and 100 % source data) than Model 2 proves that two-stage TrAdaBoost.R2 can efficiently extract knowledge from spatially outdated source data than does AdaBoost.R2. As expected, Model 3, using 70 % of target data, presented the highest prediction accuracy due to its large sample size, but the required 70 % target data limits the method's feasibility. The two-stage TrAdaBoost.R2 Model 1 exhibited the second-best performance and predicted more accurate large crash counts. Besides, Model 1 demanded an available source data set, of which distribution may differ from that of the target data, and only 30 % (about 50 segments in this study) of target data, which is reasonable sample size for calibration according to HSM.

Though the calibration factor in Model 5 reduced the prediction errors in negative binomial Model 6 in most cases, the MSE values of Models 1, 2, 3, and 4 were consistently smaller than those of Models 5 and 6. It can be concluded that the transfer learning technique achieves better predictive performance than the conventional regression model. The two-Stage TrAdaboost.R2 Model 1 is recommended to jurisdictional agencies due to its high prediction accuracy and adaptability to a small sample size.

A discrepancy of transferability in the temporal dimension was observed, which was indicated by the fact that Guangzhou data was more transferable to Shanghai data in the off-peak time period while it was less transferable in the peak time period. Thus, crash prediction models are proposed to construct in different time periods.

This study offers a base for further study of calibration methods. More research can be done to explain the mechanism of model transferability and to determine what kind of data domain tends to be transferable to another. One possible solution is to apply the transfer learning technique to measure the similarity between different data domains, which would help researchers and practitioners to develop and apply transfer models to cities where they are needed. Finally, it is worth investigating the performance of transfer learning techniques on a variety of data sets.

Declarations of interest

None.

CRediT authorship contribution statement

Dongjie Tang: Conceptualization, Methodology, Software, Data curation, Writing - original draft, Visualization, Investigation, Validation, Writing - review & editing. **Xiaohan Yang:** Conceptualization, Methodology, Software, Data curation, Writing - original draft, Investigation, Supervision, Validation, Writing - review & editing. **Xuesong Wang:** Conceptualization, Methodology, Data curation, Writing - original draft, Investigation, Supervision, Validation, Writing - review & editing.

Acknowledgments

This work was supported by the International Science & Technology Cooperation Program of China (2017YFE0134500) and the Science and Technology Commission of Shanghai Municipality (18DZ1200200). The authors are grateful to Barbara Rau Kyle for her helpful edit and thoughtful feedback.

References

AASHTO, 2010. Highway Safety Manual. American Association of State Highway and Transportation Officials, Washington, D.C.
Bauer, E., Kohavi, R., 1999. An empirical comparison of voting classification algorithms: bagging, boosting, and variants. Mach. Learn. 36 (1-2), 105–139.
Breiman, L., 1996. Bagging predictors. Mach. Learn. 24 (2), 123–140.
Brimley, B.K., Saito, M., Schultz, G.G., 2012. Calibration of highway safety manual safety performance function. Transp. Res. Rec.: J. Transp. Res. Board 2279 (1), 82–89.
Cafiso, S., Di Silvestro, G., Di Guardo, G., 2012. Application of highway safety manual to italian divided multilane highways. Procedia - Soc. Behav. Sci. 53, 910–919.
Chan, J.C.-W., Huang, C., Defries, R., 2001. Enhanced algorithm performance for land cover classification from remotely sensed data using bagging and boosting. IEEE Trans. Geosci. Remote. Sens. 39 (3), 693–695.
Dai, W., Yang, Q., Xue, G.-R., Yu, Y., 2007. Boosting for transfer learning. Proceedings of the Proceedings of the 24th International Conference on Machine Learning 193–200.
Drucker, H., 1997. Improving regressors using Boosting techniques. Proceedings of the ICML 107–115.
Farid, A., Abdel-Aty, M., Lee, J., Eluru, N., Wang, J.H., 2016. Exploring the transferability of safety performance functions. Accid. Anal. Prev. 94, 143–152.
Farid, A., Abdel-Aty, M., Lee, J., Eluru, N., 2017. Application of bayesian informative priors to enhance the transferability of safety performance functions. J. Safety Res. 62, 155–161.
Farid, A., Abdel-Aty, M., Lee, J., 2018. Transferring and calibrating safety performance functions among multiple states. Accid. Anal. Prev. 117, 276–287.
Freund, Y., Schapire, R.E., 1996. Experiments with a new boosting algorithm. Proceedings of the ICML 148–156.
Friedman, J., Hastie, T., Tibshirani, R., 2000. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). Ann. Stat. 28 (2), 337–407.
Kaaf, K.A., Abdel-Aty, M., 2015. Transferability and calibration of highway safety manual performance functions and development of new models for urban four-lane divided roads in Riyadh, Saudi Arabia. Transp. Res. Rec. 2515 (1), 70–77.
La Torre, F., Meocci, M., Domenichini, L., Branzi, V., Tanzi, N., Paliotto, A., 2019. Development of an accident prediction model for italian freeways. Accid. Anal. Prev. 124, 1–11.
Li, J., Wang, X., 2017. Safety analysis of urban arterials at the meso level. Accid. Anal. Prev. 108, 100–111.
Li, J., Wang, X., Abdel-Aty, M., Yu, R., 2018. Exploring the transferability of Cross-Country safety performance functions for Urban arterials with pooled data. In: Proceedings of the 97th Annual Meeting of the Transportation Research Board. Washington, D.C..
Lord, D., Miranda-Moreno, L.F., 2008. Effects of low sample mean values and small sample size on the estimation of the fixed dispersion parameter of poisson-gamma models for modeling motor vehicle crashes: a bayesian perspective. Saf. Sci. 46 (5), 751–770.
Pan, S.J., Yang, Q., 2009. A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22 (10), 1345–1359.
Pardoe, D., Stone, P., 2010. Boosting for regression transfer. Proceedings of the Proceedings of the 27th International Conference on International Conference on Machine Learning 863–870.
Schapire, R.E., 2013. Explaining Adaboost. Empirical Inference. Springer, pp. 37–52.
Srinivasan, R., Carter, D., 2011. Development of Safety Performance Functions for North Carolina. North Carolina. Dept. of Transportation. Research and Analysis Group.
Srinivasan, R., Colety, M., Bahar, G., Crowther, B., Farmen, M., 2016. Estimation of calibration functions for predicting crashes on rural two-lane roads in Arizona. Transp. Res. Record: J. Transp. Res. Board 2583 (1), 17–24.
Sun, C., Brown, H., Edara, P., Claros, B., Nam, K., 2014. Calibration of the Hsm's Spfs for Missouri. Publication CMR14-007 Missouri Department of Transportation.
Van Rossum, G., Drake, F.L., 2011. The Python Language Reference Manual Network Theory Ltd.The Python Language Reference Manual Network Theory Ltd.
Wang, X., Fan, T., Chen, M., Deng, B., Wu, B., Tremont, P., 2015. Safety Modeling of Urban Arterials in Shanghai, China. Accid. Anal. Prev. 83, 57–66.
Wang, X., Feng, M., Shi, Q., Li, Y., 2018a. Hotspot identification for freeways considering difference in single-and multiple-vehicle crashes. In: 97th Annual Meeting of the Transportation Research Board. Washington, D.C..
Wang, X., Yuan, J.H., Schultz, G.G., Fang, S.E., 2018b. Investigating the safety impact of roadway network features of suburban arterials in Shanghai. Accid. Anal. Prev. 113, 137–148.
Wang, X., Zhou, Q.Y., Quddus, M., Fan, T.X., Fang, S.E., 2018c. Speed, speed variation and crash relationships for urban arterials. Accid. Anal. Prev. 113, 236–243.
Xie, F., 2011. Calibrating the highway safety manual predictive methods for Oregon rural State highways. In: Proceedings of the 90th Annual Meeting of the Transportation Research Board. Washington, D.C..
Xie, K., Wang, X., Huang, H.L., Chen, X.H., 2013. Corridor-level signalized intersection safety analysis in Shanghai, China using bayesian hierarchical models. Accid. Anal. Prev. 50, 25–33.
Yu, R.J., Wang, X., Abdel-Aty, M., 2017. A hybrid latent class analysis modeling approach to analyze urban expressway crash risk. Accid. Anal. Prev. 101, 37–43.
Zhou, Z.-H., 2014. Large margin distribution learning. Proceedings of the IAPR Workshop on Artificial Neural Networks in Pattern Recognition 1–11.