Estimation and Prediction of Road Free Flow Speed with More Efficient DNN Map Matching Results

Shunlei Ning
Heng Liu
Songjian Zhang
DiDi Chuxing
Beijing, China
ningshunlei@didiglobal.com
aidanliuheng@didiglobal.com
zhangsongjian@didiglobal.com

Yuqin Jiang
Jingbo Han
Shui Liu
Jun Fang
DiDi Chuxing
Beijing, China
jiangyuqin@didiglobal.com
hanjingbo@didiglobal.com
liushui@didiglobal.com
fangjun@didiglobal.com

Naiqiang Tan
Hua Chai
Bo Zhang
DiDi Chuxing
Beijing, China
tannaiqiang@didiglobal.com
chaihua@didiglobal.com
zhangbo@didiglobal.com

ABSTRACT

Route planning is the key component of the industry navigating engine system, which actually consists of two core parts—the planning algorithm based on graph theory and the statistical links' travel time estimation. In recent years, the first part has attracted more and more academic attention and many excellent algorithms such as CH, CRP, CCH have emerged. However, the second part is rarely studied, and there is no excellent solution so far. In this article, we try to estimate the free flow speed of the road segment from the raw GPS collections to solve the second problem. In practice, the types and status of links are complicated. For example, for the links within the same road, some links have more turns, while some are adjacent to the crossing of the road with traffic light. These differences lead to the variance of the through capacity. In the worst case, some links may have fewer historical GPS points or some links are always congested so that we can't study them directly. In our paper, we propose a systematic solution, which includes 3 main parts: 1. we will introduce a more efficient Map Matching algorithm, comparing to the classical Hidden Markov framework, showing greater than 8% improvement of accuracy; 2. for road segments which have adequate history GPS points, we use Bayesian Non-parametric Model(DPMM, Dirichlet Process Mixture Model) with Markov Chain Monte Carlo (MCMC, Collapsed Gibbs sampling) sampling to estimate the free flow speed; 3. for the rest of links, we develop sophisticated machine learning algorithms for fewer GPS trajectories coverage. The above methods have been applied in our routing service and work well all the time.

KEYWORDS

DNN, Map Matching, Free Flow Speed, DPMM, Semi-supervised Regression

Conference'28, August 2022, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Shunlei Ning, Heng Liu, Songjian Zhang, Yuqin Jiang, Jingbo Han,, Shui Liu, Jun Fang, Naiqiang Tan, Hua Chai, and Bo Zhang. 2022. Estimation and Prediction of Road Free Flow Speed with More Efficient DNN Map Matching Results. In *Proceedings of ACM Conference (Conference'28).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

How to estimate the link' travel time(or speed, length divides speed can get the link' travel time) of the real world road is an important problem, it is the basics of most useful modern traffic techniques such as estimating travel time of arrival(eta), traffic condition estimation, ranking(for paths), especially route planning. Routing planning algorithm based on graph theory has developed rapidly in the past 20 years. Many excellent algorithms such as CH[9], CRP[6], CCH[7] can calculate the optimal route in milliseconds even with large continental road networks. However, proper links' weight(link travel time) are equally important with the planning algorithms for once reasonable route planning request.

In our study, we collect 2 months of high quality of GPS trajectories of DiDi Chuxing's online car hailing orders in Beijing. We develop a set of innovative and reliable solution from Map Matching to statistical estimation and sophisticated machine learning prediction of FFS (free flow speed). In Chapter 2, we will introduce our more efficient DNN Map Matching Algorithm, comparing to the classical Hidden Markov framework, showing greater than 8% improvement of accuracy. In Chapter 3, for GPS data of adequate collections, we present the estimation of FFS by DPMM(Dirichlet Process Mixture Model) with collapsed Gibbs sampling. In Chapter 4, we will present the union of Supervised and Semi-supervised machine learning algorithms for road with less GPS collections. In Chapter 5, we will put the outstanding experiment results up. Figure1 shows the flow diagram of our works. Table 1 summarizes the terminologies used in this paper.

To sum up, the contributions of the paper are as follows:

- The first time using DNN incorporating spatial features in map matching algorithm, which shows over 8% improvement of accuracy when compared with HMM
- Systematical solution to get link's proper free flow speed with the Bayesian Non-parametric model, supervised model

and using semi-supervised algorithm to promote the regressive effect.
- The proposed method has been deployed in a real system and is shown to be effective.

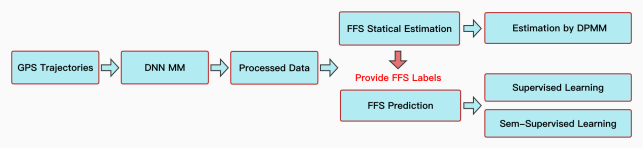


Figure 1: Flow Diagram

Table 1: Terminologies used in this paper

Term	definition	notation
Road Network	A directed, weighted graph describing the road relations in real world	G(V, L)
Link	An edge in the road network, representing a road segment	L_i
GPS Trajectory	a sequence of time-stamped points, each of which contains the information of latitude, longitude and instantaneous speed	p_i^n
Free Flow Speed	Operating speed on a link in free-flow state	FFS
Map Match	Match recorded geographic coordinates to a road of the real world	MM

2 MAP MATCHING ALGORITHM

Map matching is the problem of how to match recorded geographic coordinates to a road of the real world. As shown in Figure 2, the dots are the GPS trajectory of the vehicle, and the blue curve is the corresponding road. Early research on map matching can be divided into three categories: geometric topology methods[3, 11], statistical probability methods[12, 15, 17, 19, 26, 28], and other advanced techniques[10, 20, 29]. As early as 2009, Paul Newson and John Krumm developed the well-known hidden Markov analysis framework[19]. At the same time, an almost similar method ST-MATCHING has also been developed by Yu Zheng[15], which uses spatial analysis, temporal analysis and dynamic programming algorithms to perform map matching. Later, many novel methods were proposed, such as the IVMM algorithm[28] which is based on the idea of interactive voting, and the method based on path prediction proposed by Hu[26]. In recent years, many machine learning algorithms have also been applied to map matching. Takayuki[20] used inverse reinforcement learning to improve the accuracy of transition probability. In addition, Kai Zhao proposed DeepMM[29], which uses serialized modeling and uses the attention mechanism to improve the effect of map matching.

Figure 2: Map Matching Example

This paper proposes a new map matching method based on deep learning. Different from the previous map matching methods, we introduce road-attributes information in the road network and point-related information collected by GPS into map matching algorithm. Comparing to the classical HMM framework[19], we make some improvements, not only including the simple spatial information, we also introduce some road attributes to improve the accuracy of the transit and observation probability, such as speed, number of lanes, road speed limit, etc. In addition, when vehicles pass the same road section, there will be similar driving patterns, which contain a lot of useful information for map matching, such as driving direction and observation deviation... We can obtain these information through data mining techniques so that our model can complete the map matching task more accurately.

Based on the above data, we designed a 7-layers feed-forward neural network. Compared to the traditional models, the deep neural network has excellent feature interaction capabilities, and it can well capture the hidden information between the GPS points and the road. Finally, in order to prevent overfitting, batch normalization is used between each layer. We update the parameters of the model by performing gradient descent to minimize the cross-entropy loss function. The model structure is shown in Figure 3.

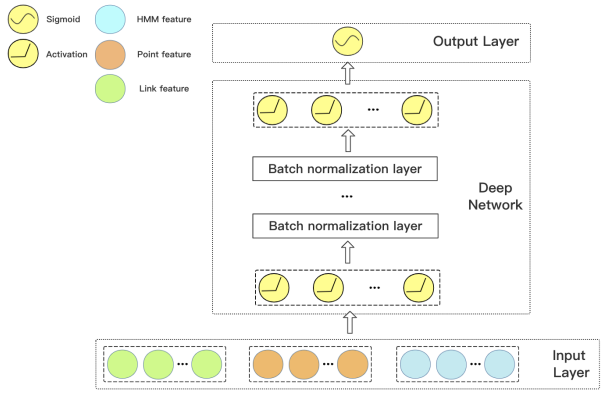


Figure 3: DNN Map Matching Structure

3 FFS ESTIMATION

Lacking of real-time road traffic flow information, it is difficult to estimate the free flow speed (Abbreviated as FFS) of the road, based on the original definition of free flow by using online car-hailing trajectory data. To get the estimation of FFS with original GPS data set, we makes the following assumption:

Assumption 3.1. *The operating speed V_i on the road L_i is a random variable on $\mathbb{R}^+$, which confirms to a certain probability distribution $f_i \in \mathcal{F}$. In an ideal environment (no extreme weather , traffic accidents), the distribution of V_i is only related to the traffic conditions on the road (free flow, amble, congestion and extreme congestion). Under a given traffic condition state z_k, V_i confirms to a fixed probability distribution f_{i,z_k}. That is,*

$$V_i \sim f_{i,z_k}(v), k = 1, \ldots, i_K$$

Based on the above assumption, we propose a set of algorithms to estimate the FFS of all links in the road network using GPS trajectory data. At first, this study maps the raw historical trajectories to the road network of the digital map through the map matching algorithm introduced in Chapter 2. After certain data processing and calculations, the historical operating speed distribution of each road is obtained. Finally, the DPMM algorithm is used to separate the speed distribution in the free flow state from the historical distribution and then estimate the FFS of the road based on its mean value.

Due to the sparsity of the historical trajectory of online car-hailing, it is difficult to obtain the FFS estimation of all links in the entire road network based on trajectory data alone. In this study, we use machine learning methods to learn the relationship between link features and FFS labels set produced by the statistic inference algorithm.

The transit time of a road segment is researched for a long period of time. With the rapid development of GPS technology, the method of obtaining traffic data by using GPS based probe vehicle system has attracted extensive attention in recent years, among which estimating road transit time is one of the hot topics. Hunter[13] used the traffic data collected by the test vehicle with GPS device to estimate the transit time between adjacent intersections. Pan[21] proposed an algorithm for estimating road transit time and intersection delay based on GPS data. Firstly, GPS points are matched to the corresponding road section through a map matching algorithm, and then the road transit time and intersection delay are estimated by calculating the time difference between GPS points in and out of the road section. Pu[24] confirmed that in the free flow state, the road transit time distribution is approximately Gaussian distribution, while in the case of congestion, the data presents an obvious skew distribution, and the lognormal distribution is closer to the real distribution of the data. With the increase of the amount of data, the distribution of road transit time not only shows skewness, but also has problems of multimodal and excess kurtosis. The single probability distribution is not enough to describe the heterogeneity of road transit time distribution. The research shows that[4], compared with the single distribution model, the mixed distribution model can better fit the road transit time distribution. Park[23] proposed a multi state model based on Gaussian mixture distribution to fit the transit time distribution of highway.

In this study, we use the Bayesian non-parametric model DPMM (Dirichlet Process Mixture model) to model the historical speed distribution. The method of pre-setting the number of fixed sub-distributions is not applicable based on the parametric model, since the proportions of different road conditions on different roads are not consistent(for example, some remote roads rarely appear congested or unblocked all the time). Compared with the traditional finite mixture model (FMM), such as the Gaussian mixture model (GMM) whose base distribution is the Gaussian distribution, DPMM adaptively learns the number of sub-populations in the mixed distribution, and can better fit the true distribution of the data, which is validated by the following experiments 4. A large number of studies have shown that the road speed distribution confirms to the normal distribution or the mixed normal distribution[8, 14, 22]. Therefore, we select the normal distribution as the base distribution of the DPMM, and use the collapsed Gibbs sampling method to infer the DPMM. In particular, we take the maximum mean of the sub-populations in the result as the FFS.

DPMM's inference methods can be roughly divided into two categories: the first is an algorithm based on Monte Carlo Markov chain sampling, of which the most classic algorithm is an inference algorithm based on Gibbs sampling[18]. This method is relatively intuitive, but its calculation is relatively slow facing large-scale data set; the second one is the Variational Bayesian inference algorithm[1], using a specific lower bound of the integral for approximation, is more computationally efficient. However it is also easy to fall into the local optimal trap. In this paper, we used an inference method based on collapsed Gibbs sampling of MCMC(Markov Chain Monte Carlo) to get the most accurate estimation of links' FFS, which refers to the Algorithm 6 in [27].

In the specific inference process, due to the poor computational efficiency of the Gibbs sampling method facing large-scale data, while the data volume of some links in the historical data can reach the level of 100,000, the memory and time consumption are relatively large when using complete data to infer. To solve this problem, we tried different methods. The effective one is to perform multiple random sampling on the original data set, and take the average of multiple results as an estimate of the FFS in algorithm 1.

4 FFS PREDICTION

For some links with a small number of historical trajectories, the estimation based on the historical speed distribution is not reliably enough, not to mention that most links have almost no historical trajectory coverage. Therefore, we firstly extract the FFS of "hot" links as the true label. And then based on this part of the labeled data set, we train the machine learning model to predict the FFS of "unpopular" links. As the amount of historical trajectory data is larger, the inference based on the mixed distribution model is more accurate. Based on this fact, according to the amount of historical trajectory data from more to less, we divide the data set into three parts: labeled data set, weakly labeled data set and unlabeled data set.

4.1 Supervised Learning

When there is not sufficient data to conduct an accurate estimation of FFS, we switch to another perspective that treating the FFS

Algo 1: FFS-DPMM

Data: all of $link_i$ GPS speed collection $V_i = \{v_k\}_{k=0}^{n_k}$
Input: number of iteration n, number of sample size m, confidence threshold τ
Result: FSS_i
if $n_k \geq 8000$ **then**
  for $j \leftarrow 1$ **to** n **do**
    $V_{ij} = sample(V_i, m)$
    model= Collapsed_DPMM.fit(V_{ij})
    // Collapsed_DPMM refers to the Algorithm 6 in [27]
    μ = model.means
    w = model.weights
    for μ_l, w_l *in* μ, w **do**
      if $w_l < \tau$ **then**
        $\mu = \mu.delete(\mu_l)$
      end
    end
    FFS_{ij} = max(μ)
  end
  $FFS_i = \frac{\sum_{j=1}^{n} FFS_{ij}}{n}$
end

Table 2: Division of Data Set

Sample Size	Data Set	Percentage
>5000	Labeled Data Set D_L	10.75%
100-5000	Weakly Labeled Data Set D_W	25.04%
<100	Unlabeled Data Set U	64.21%

estimation as a regression problem rather than a statistical inference problem. We then select 54-dimension link features as the model input, such as road width, speed limit, the number of lanes, etc. Naturally, the label is the FFS produced by DPMM. Since the features have a large proportion of categorical variables and some features are relatively sparse, we select XGBoost[5] regression model as the supervised model.

However, because the labeled data only accounts for a limited number of links, and it can be seen from the Figure 6 that the statistical frequency distribution(not absolute quantity) of the labeled data set is not consistent with the distribution of the overall data set. Some of the link features have the problem of sparseness, so the direct use of supervised learning to predict unlabeled data will lead to larger deviations in the accuracy of the results. In the two-month GPS trajectory data in Beijing, the proportion of links covered by at least one historical trajectory in the entire Beijing road network is 82.25%, and the proportion of links covered by more than 100 historical trajectories is only 35.79%. Since many inner roads of some communities or companies are not allowed to be reached by online car hailing, the trajectories sparsity is reasonable in the case of online car hailing services. To make use of information of unlabeled data, we attempt to combine the information in labeled data and unlabeled data for learning to improve the overall performance of the model and reduce the model deviation caused by the distributions inconsistency.

4.2 Semi-supervised Learning

For supervised learning based only on labeled data, the model prediction still has a large margin of error. Furthermore, due to differences in distribution between labeled data sets and overall data sets, the supervised model has poor generalization abilities in unlabeled data sets. This section attempts to use the idea of semi-supervised learning, combined with labeled data and unlabeled data for model training to improve the performance of the model. In this study, two classic semi-supervised learning methods were selected: self-training and collaborative training.

Self-training[25] is one of the most common algorithms used in semi-supervised learning. When the training set includes only a small amount of labeled data and a large amount of unlabeled data, the self-training paradigm iteratively trains model to augment labeled training data: At each iteration, a supervised model (Such as linear regression, XGBoost, etc.) is trained using labeled data set. According to certain standards, put the unlabeled samples with high prediction confidence into the labeled data set, using its predicted result as its "pseudo-label". It loops until there is no confident unlabeled data available. The disadvantage is that it relies heavily on the accuracy of the initial model. Once the model predicts an error, its error will gradually accumulate with the self-training process. Therefore, when there are outliers in the data set or the data is unevenly distributed, self-training algorithm tends to have poor performance, or even worse than a purely supervised learning model.

Self-training algorithms[16] are often applied for classification. Usually based on Bayesian theory, self-training algorithms use the posterior probability of the model to predict each category to determine the model's confidence in the prediction of unlabeled data, and can be based on the proportion of each category in the training set. Adjust the proportion of categories that add "pseudo-label" data in each iteration to alleviate the problem of data imbalance. For regression problems, since the output of the model is a continuous variable, it is difficult to find a reliable method to determine the confidence of the "pseudo-label". Based on the manifold hypothesis of semi-supervised learning, this study combined with the labels of the nearest neighbors in the labeled data set, to provide a basis for judging the confidence of the "pseudo-label" produced by the model. The specific criterion for such a judgment is:

$$L_1 = (KNN\_Regressor(x_u) - XGB\_Regresssor(x_u))^2$$

In addition, considering that some of the features in the labeled data set are very sparse, and their distribution is quite different from that of the unlabeled data set, simply using the nearest neighbor method for confidence judgment may cause the model to have large prediction errors for such unlabeled data. In this case, this part of the data set either failed to pass the confidence judgment, or the KNN regressor and XGBoost had the same direction deviation and were pseudo-labeled. Errors may continue to accumulate during the iteration process of self-training, resulting in poor performance of the model. Therefore, when judging the confidence levels, it is necessary to minimize the error caused by this type of data. In this study, due to the particularity of the data itself, some unlabeled data have "weak label" with noise: the estimated value refered by DPMM algorithm with insufficient data. This part of the information

can be used to make certain confidence judgments on the "pseudo-labels" predicted by the model. Under the dual supervision of "weak label" and nearest neighbor label data, the model's judgment of confidence is more reliable. For weak label data x_u, the criterion for using pseudo-labels to judge the confidence is:

$$L_2 = (weak\_label(x_u) - KNN\_Regressor(x_u))^2$$

$$L_3 = (weak\_label(x_u) - XGB\_Regresssor(x_u))^2$$

Based on the above analysis, this article sets the confidence judgment method as: when any two of the three losses are less than a certain threshold at the same time, add it into the training set with its pseudo-label, which is the mean of the three regressorsṕrediction, and record this model as STT (self-tri-training) in algorithm 2.

Algo 2: Self-Tri-Training

Data: Dataset L : links' features with FFS label
U = links' features without FFS label
τ: the confidence threshold
repeat
  $Model \leftarrow XGBoost(L)$
  for $x \subset U$ **do**
    if $||Model(x) - DPMM(x)|| \leq \tau$ *or*
    $||Model(x) - KNN(x)|| \leq \tau$ *or*
    $||KNN(x) - DPMM(x)|| \leq \tau$ **then**
      pseudo_label = $\frac{Model(x)+KNN(x)+DPMM(x)}{3}$
      $L \leftarrow L \cup (x, pseudo\_label)$;
      $U \leftarrow U/\{x\}$
    end
  end
until *No Updates For L*;

Co-training[2] is a classic semi-supervised learning algorithm based on divergence. Its core idea is to train two learners separately based on two fully redundant views in the multi-view data. Co-training greatly utilizes the information of unlabeled data through the way of two learners labeling each other with "pseudo-labels". Its effect has also been theoretically proven with a strict data hypothesis, that is, there are two fully redundant and mutually conditionally independent views in the data set. Full redundancy means that a sufficiently good learner can be trained based on any one of views. And mutual conditional independence means that the two views are independent of each other under the conditions of a given result. In practical applications, it is often difficult to meet this hypothesis. However, many studies have shown that collaborative training still has a relatively good effect, even when the data does not meet the strict hypothesis.

When collaborative training was proposed, it was a semi-supervised classification algorithm, but its ideas are also applicable to regression models. Zhou[30] proposed a collaborative regression algorithm (COREG), which uses two regressors to generate pseudo-labeled data for each other, and the confidence of the "pseudo-label" is based on the mean square error reduction of its neighborhood in the training set. The theoretical basis of this method is the manifold hypothesis that the samples in the tiny neighborhood have similar properties or labels. COREG cleverly uses a different number of nearest neighbors and different distance measures to construct differences between learners. Analysis and experiments show that COREG can effectively use unlabeled data sets to improve the performance of the regressor.

In addition, we also tried a COREG model, using two different KNN regressors to filter unlabeled data sets, and combining weak labeled data sets to improve the confidence judgment for pseudo-labels COREG_REV in algorithm 3. The original paper first uses a KNN regressor to predict unlabeled data, and compares the mean square error of the labeled data in the neighborhood before and after adding the pseudo-label to determine the confidence level. When there is a difference in distribution between the labeled data set and the unlabeled data set, many unlabeled data have a small similarity between the nearest neighbors in the labeled data. As a result, using the predicted value of the KNN regressor directly used as its pseudo label could reduce the performance of the final regression model. Therefore, in practical applications, when the average distance between the unlabeled data and its nearest neighbors is greater than a certain threshold, the nearest neighbor selection range of the KNN regressor is extended to the weak-labeled data pool (including the currently predicted unlabeled data).

5 STUDY DATA AND EXPERIMENT

5.1 Dataset

The data set used in this article includes topological structure data of the road network in Beijing, the corresponding link features, and the GPS trajectory data of the car-hailing order from February 1 to March 31, 2021.

Road network($G(V, L)$) is a form of transforming real-world roads into a topological network structure. Based on certain standards, key points in a road will be mapped into a road network as the node, and the directed road between adjacent nodes corresponds to the link in the road network. In addition, the road network data also includes the road attributes corresponding to each link and the longitude and latitude information of the nodes contained in the link.

The GPS trajectory data used includes the GPS trajectory stream produced by the car-hailing GPS device. The sampling frequency is $2 \sim 4s$. Each GPS data includes the latitude, longitude, instantaneous speed and other information of the GPS point at that point.

5.2 Trajectory data processing

After the map matching algorithm, we can obtain the location and speed information corresponding to each GPS point. But in order to obtain an accurate historical road speed distribution, we still need to remove some noise data:

(1) Map matching GPS points with low confidence level. Due to signal problems or subjective factors of the driver, there are amounts of trajectory missing or GPS drafting points in the trajectory data. In the process of data processing, it is necessary to remove these GPS points with low confidence level in map matching.
(2) GPS points where the vehicle stays abnormally. For the GPS trajectory where the vehicle stays abnormally, a large number of GPS points with the same position information and speed of 0 will be returned. If they were not removed, it will

Algo 3: COREG_REV

Input: L = {Links with FFS labels | (X, Y)},
L_w = {Links with weak FFS labels | (X_w, Y_w)},
U_u = {Links without any FFS labels | X_u},
U = {Links without FFS labels | $X_w \cup X_u$},
maximum number of learning iterations T,
num of nearest neighbors k_1, k_2,
sample size s,
distance metrics D_1, D_2.

Result: FFS of U

PROCESS:

$L_1 \leftarrow L; L_2 \leftarrow L$
$U_p \leftarrow sample(U, s)$;
$h_1 \leftarrow kNN(L_1, k_1, D_1)$;
$h_2 \leftarrow kNN(L_2, k_2, D_2)$;

repeat
  for $j \in \{1, 2\}$ **do**
    for *each* $x_u \in U_p$ **do**
      $\Omega_u \leftarrow Neightbors(x_u, L_j, k_j, D_j)$
      // the revised part of COREG_REV
      if $mean\left(D_j(x_u, \Omega_u)\right) > \tau$ **then**
        $\Omega_u \leftarrow \text{Neighbors}\left(\mathbf{x}_u, L_j \cup L_w, k_j, D_j\right)$
      end
      // the other part refers to COREG[30]
      $\widehat{y_u} \leftarrow h_j(x_u)$
      $h_j^{'} \leftarrow kNN(L_j \cup \{x_u, \widehat{y_u}\}, k_j, D_j)$
      $\delta_{x_u} \leftarrow \sum_{x_i \in \Omega_u}((y_i - h_j(x_i))^2 - (y_i - h_j^{'}(x_i))^2)$
    end
    if $\exists \delta_{x_u} > 0$ **then**
      $\widehat{X}_j \leftarrow argmax_{X_u \subset U^{'}} \delta_{X_u}; \widehat{Y}_j \leftarrow h_j(\widehat{X}_j)$
      $\pi_j \leftarrow \{(\widehat{X}_j, \widehat{Y}_j)\}; U^{'} \leftarrow U^{'} - \{\widehat{X}_j\}$
    end
  end
  $L_1 \leftarrow L_1 \cup \pi_2; L_2 \leftarrow L_2 \cup \pi_1$
  $\pi_1 \leftarrow 0$; $\pi_2 \leftarrow 0$
  $h_1 \leftarrow kNN(L_1, k_1, D_1)$;
  $h_2 \leftarrow kNN(L_2, k_2, D_2)$;
  $U_p \leftarrow sample(U, s)$
  if *neither* L_1 *and* L_2 *changes* **then**
    exit
  end
until *T Rounds*;
$f_1 \leftarrow XGBoost(L_1)$;
$f_2 \leftarrow XGBoost(L_2)$
return $\frac{1}{2}(f_1(U) + f_2(U))$

have a greater impact on the historical speed distribution of the road.

(3) GPS trajectories related to the start and end points of the online car-hailing order. Due to its commercial nature, the initial part of each order trajectory data cannot accurately reflect the road operating speed distribution, so the 200m trajectory points connected to the start and end of orders are removed during data processing.

(4) GPS points with abnormal speeds due to GPS information errors or subjective speeding by the driver. This article randomly selects some samples with abnormal speeds. By analyzing this part of the samples, GPS points with instantaneous speeds that exceeds the link speed limit 1.2 times is screened out based on experience.

5.3 Map Matching Effect Comparison

The evaluation index is accuracy, and the specific definition is as follows:

$$accuracy = \frac{P_{sm} \cap P_{sg}}{P_{sg}}$$

where *Psm* is the result after the observation point is matched and *Psg* is the ground truth (i.e. the "true" path of the vehicle) of the observation point.

We compare the performance of our model with HMM(considering HMM as the baseline, not only it has been tested, but also it meets the real-time performance of industrial applications). The performance comparison results are shown in Table 3. Under our DiDi Chuxing's 10w+ map matching labeled dataset, compared with HMM, the map matching algorithm based on deep learning proposed in this paper improves the accuracy by 8.08%. The reason can be concluded that with the help of a large amount of auxiliary information, the deep neural network can better capture the implicit semantics between the observation point and the road. It is worth mentioning that this method is used in Didi's online environment.

Table 3: The performance of our model and baselines

Model	HMM	DNN_MM
Accuracy	0.8836	**0.955**
Improve	0%	**+8.08%**

5.4 Calculation of Link Transit Speed

With map match results, we need to calculate the link transit speed based on GPS trajectory. The specific calculation process is as follows: assuming a trajectory $\mathbf{T} = traj(l_i, v_i)_{i=0}^{n}$, where l_i is the road network link matched by GPS points, v_i Is the instantaneous speed of the GPS point. Let the set of trajectory points matching l_i be I, the length of l_i is L_i, then the transit speed of the link is:

$$v_i = \frac{\sum_{k \in I} v_k}{|I|}$$

5.5 FFS Estimation By DPMM

According to the above process, we can get all the historical traffic speeds of the trajectory passing through the link. In this study, the Bayesian non-parametric model DPMM is used to model the historical speed distribution. In particular, we select Gaussian as the base distribution of DPMM.

In the specific inference process, considering calculation performance, this article randomly sampled the original data set several times, and took the average of the multiple sampling results as an estimation of the FFS. As for the sample size setting of random sampling, we randomly selected 100 links with a data volume greater

than 20,000, and used the grid search method to find the optimal sampling number. It was found that when the sample size was greater than 5000, the final result was close to convergence. The error between the results obtained from the original data does not exceed 0.2m/s, and the estimated variance is also less than 0.5. On the basis of comprehensive consideration of performance and error, we set the final sampling size to 8000, the number of sampling times to 5, and the inference result of links with a data volume greater than 5000 as the true label of their FFS, as shown in Figure 4.

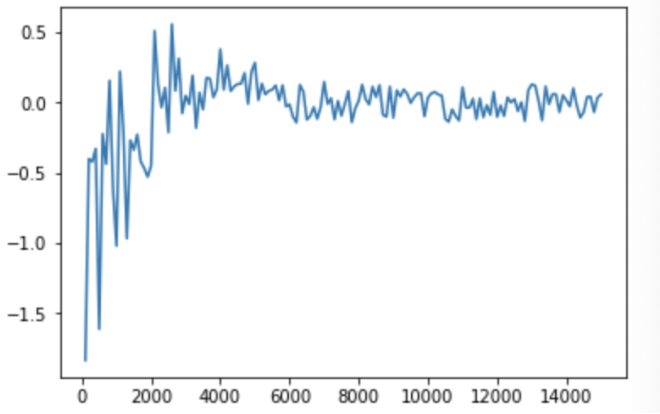


Figure 4: Deviation of DPMM and Sampling Quantity (x-axis represents the difference of sampling quantity and the y-axis represents the deviation of DPMM inferring result)

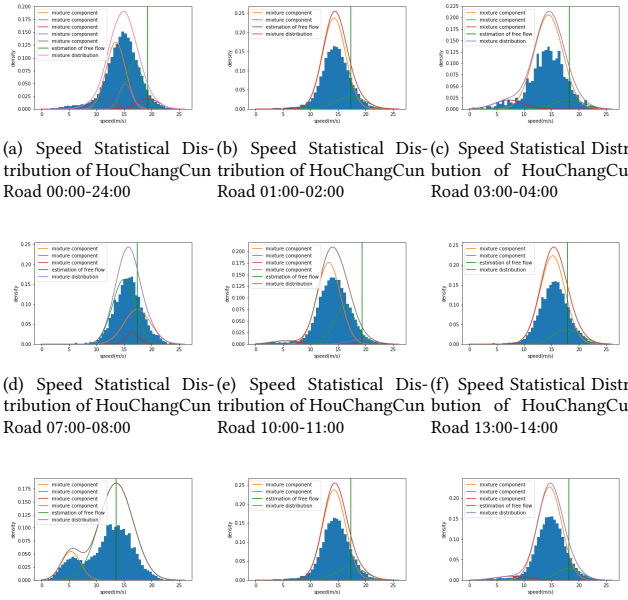
In order to prevent a small amount of noise data from affecting the estimation accuracy, only sub-distributions with a weight greater than a certain threshold are selected in the actual estimation process. For the selection of the weight threshold in the algorithm, this paper sets it to 0.05 based on experience. In addition, the initial maximum number of clusters is set to 10, and the relevant prior distribution of the parameters is set to: $\alpha \sim \Gamma(1,1), \mu_k \sim N(\mu,1), \sigma_k^{-2} \sim \Gamma(1,1)$, where μ is the mean value of the overall data distribution, the maximum number of iterations of gibbs sampling is 10000, and the first 5000 times are the burn-in stage.

This paper uses three evaluation methods: KS test, KL divergence, and JS divergence. Based on randomly selected 10,000 links, the fitting effects of DPMM and GMM with different prior parameter settings are compared. The results show that, in any evaluation method, the fitting effect of DPMM on historical velocity data sets is better than GMM. In the inference result of DPMM, the average number of sub-distributions is 2.50, the maximum number of sub-distributions is 5. The proportion of data with the numbers of 2 and 3 is 87.2%. The specific evaluation results are shown in the table 4.

According to the collapsed DPMM algorithm, we randomly selected the historical speed distribution of some links, and analyzed the mixed distribution under different time periods. As shown in the result in the Figure 5, this figure shows the historical speed distribution of the link in the section of Houchangcun Road in Beijing at different times. In different times, the speed meets different mixed distributions. During peak periods, The road conditions are more congested, and the mixed distribution will include a sub-distribution

Table 4: Comparison of data fitting degree between DPMM and GMM with different parameters

Model	K-S test *pvalue*	KL divergence	JS divergence
DPMM	**0.3476**	**0.3400**	**0.0701**
GMM(K=2)	0.1127	0.4399	0.0909
GMM(K=3)	0.1622	0.4645	0.8594
GMM(K=4)	0.2194	0.4510	0.0845
GMM(K=5)	0.3387	0.4227	0.0817
GMM(K=6)	0.3373	0.4150	0.0809



(a) Speed Statistical Distribution of HouChangCun Road 00:00-24:00
(b) Speed Statistical Distribution of HouChangCun Road 01:00-02:00
(c) Speed Statistical Distribution of HouChangCun Road 03:00-04:00
(d) Speed Statistical Distribution of HouChangCun Road 07:00-08:00
(e) Speed Statistical Distribution of HouChangCun Road 10:00-11:00
(f) Speed Statistical Distribution of HouChangCun Road 13:00-14:00
(g) Speed Statistical Distribution of HouChangCun Road 18:00-19:00
(h) Speed Statistical Distribution of HouChangCun Road 21:00-22:00
(i) Speed Statistical Distribution of HouChangCun Road 23:00-24:00

Figure 5: Different Speed Statistical Distribution of HouChangCun Road

with a lower mean speed, and during unblocked periods, the sub-distribution with a higher mean speed has a higher weight. Except for the two peak periods of 7 a.m. and 6 p.m., the FFS estimated by DPMM under different time periods. This also verifies the correctness of the hypothesis and the effectiveness of the algorithm for free flow estimation from one angle. During the peak period, due to the large traffic flow and the small proportion of the free flow state, it is difficult to obtain the true estimated value through DPMM. Therefore, the historical speed distribution of link throughout the day is selected for inference in practical applications.

According to the above process, we have obtained the free flow speed estimation value of some links and used it as the training set for subsequent model training. However, the data set labeled in this part only occupies 10.75% of the total number of links in the road network, whereas close to the data of 90% is difficult to obtain a reliable free circulation time estimate through DPMM.

5.6 FFS Prediction

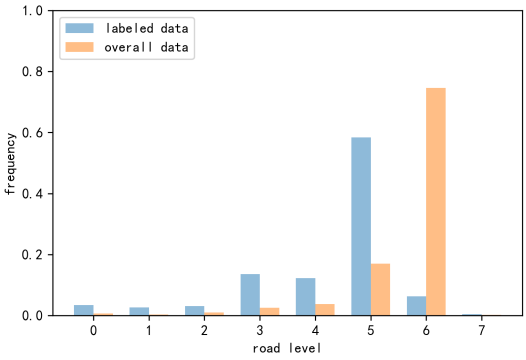


Figure 6: Comparison of road level frequency distribution(not absolute quantity) between labeled data and overall data

5.6.1 Supervised Learning. We first use the supervised learning model to predict the FFS of links with sparse historical trajectory. Based on the four evaluation indicators of *mape*, r^2, *mae* and *rmse*, several commonly used regression models are compared. The results of table 5 show that under the same condition, XGBoost's r^2, *mae* and *rmse* are better than other models, while the *mape* of MLP is slightly lower than XGBoost. In terms of the specific tuning of the parameters, we use the grid search method based on 10-fold cross-validation.

Table 5: Comparison of the performance of different supervised learning models

models	mape	r2	mae	rmse
XGBoost	0.1888	**0.6877**	**1.9059**	**2.5298**
Linear Regression	0.2135	0.6064	2.1424	2.8404
Random Forest	0.1976	0.6497	2.0116	2.6794
MLP	**0.1866**	0.6799	1.9284	2.5614

We also tested the impact of data distribution differences and some sparse features on the model. This paper first standardized the labeled data. In the cross-validation process, the Kmeans method was used to cluster the test set. The result found that the *rmse* of the XGBoost model on minority classes was 3.5222, which was higher than the average error of the test set by 39.23%. After the experiment, the characteristics of these categories are randomly evaluated manually, and it is found that most of them are roads within buildings or roads near the station, and the DPMM estimation is in line with expectations. In addition, this section uses the trained XGBoost model to predict the weakly labeled data set. It is found that the *rmse* is 3.7228, which is 47.16% higher than the test set, and *mape* is 48.11%, which is 154.82% higher than the test set. Although the labels of the weakly labeled data set are not reliably enough, the high prediction bias of the model on the weakly labeled data shows to a certain extent that inductive learning cannot be directly used to predict the unlabeled data set.

5.6.2 Semi-supervised Learning. In this study, we select two classic semi-supervised learning methods: self-training and collaborative training, and improves the model based on the particularity of the data.

In specific experiments, this paper uses cross-validation to set specific thresholds for confidence judgment. In the use of unlabeled data set, first use weakly labeled data with a historical data volume greater than 1000 for the first stage of semi-supervised learning, then perform semi-supervised learning on the remaining weakly labeled data, and finally use the unlabeled data set. This hierarchical structure is mainly to prevent the "pseudo-labels" that are first added to the labeled data set from causing large inductive biases on the model. This section conducts self-training models that use different confidence standards. Comparative experiments include the ST_1, ST_2, ST_3 model and the STT model that use the most confidence standards of L_1, L_2, L_3, and the specific experimental results are shown in Table 6.

The experimental results show that compared with the supervised learning model, the STT model has a slight improvement in the test set effect. *rmse* is reduced from 2.5298 to 2.4674, and *mape* is reduced from 0.1888 to 0.1794, the highest in the minority class in the test set *rmse* dropped from 3.5222 to 3.1202, and its performance on the weakly labeled data set has improved significantly. *rmse* has dropped from 3.7228 to 3.0247, and *mape* has dropped from 0.4811 to 0.3454. The effects of other self-training models are worse than the STT model, and the performance of the ST_2 model on the test set is even worse than that of the pure supervised learning model.

Table 6: Semi-supervised regression model comparison experiment

Modes	rmse	mape	$rmse_M$	$rmse_{WL}$	$mape_{WL}$
STT	2.4674	0.1794	3.1202	3.0247	0.3454
ST_1	2.5285	0.1853	3.1748	3.2321	0.3627
ST_2	2.5396	0.1904	3.3617	3.4867	0.4254
ST_3	2.5215	0.1861	3.1255	3.5352	0.4312
COREG	2.3355	0.1779	3.0056	3.0431	0.3329
COREG_REV	2.2475	0.1670	2.8997	2.9263	0.3154

The COREG_REV model is compared with the original COREG model. The experimental results are shown in Table 6. In terms of specific parameter settings, this paper selects two KNN regressors to filter the weakly labeled and unlabeled data sets, the number of nearest neighbors is set to 3, and the distances are respectively selected by Minkowski distance with $p = 2$ and $p = 5$. The data volume of the unlabeled data pool in each iteration is set to 10000, and the final regression adopts the XGBoost regression model with fine tuning parameters.

The experimental results show that the COREG_REV model is better than the self-training model and the vanilla COREG model in the prediction error of the test set. Compared with the supervised learning model, *rmse* is reduced from 2.5298 to 2.2475, and *mape* is reduced from 0.1888 to 0.1670, the highest *rmse* in the minority

category in the test set dropped from 3.5222 to 2.8997. The performance on the weakly-labeled data set was slightly better than the self-trained model, *rmse* dropped from 3.7228 to 2.9263, *mape* decreased from 0.4811 to 0.3154.

6 CONCLUSION AND FUTURE WORK

In this paper, we develop a set of solution for road free flow speed mining. In stage 1, comparing to traditional Hidden Markov framework, the novel algorithm of DNN map matching has a more than 8% improvement of accuracy. In stage 2, for road segments which has adequate GPS collections, we use Bayesian Nonparametric Model DPMM with Collapse Gibbs sampling to infer every link's speed statistical distribution. In stage 3, with the stage 2's free flow speed labels, we develop a sophisticated statistical machine learning algorithm union for the road segments of less GPS conditions. The above methods have been applied in our routing service and work well all the time.

The later two interesting and potential topics one is the mining work of more accurate through cost of different road corners with lights or not, another is the statistical relations between the level of congestions and road speed distributions, we will do more research in our future work.

REFERENCES

[1] David M Blei and Michael I Jordan. 2006. Variational inference for Dirichlet process mixtures. *Bayesian analysis* 1, 1 (2006), 121–143.
[2] Avrim Blum and T.M Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*. ACM, New York, 92–100.
[3] Daniel Chen, Anne Driemel, Leonidas J Guibas, Andy Nguyen, and Carola Wenk. 2011. Approximate map matching with respect to the Fréchet distance. In *2011 Proceedings of the Thirteenth Workshop on Algorithm Engineering and Experiments (ALENEX)*. Society for Industrial and Applied Mathematics, New York, 75–83.
[4] Peng Chen, Kai Yin, and Jian Sun. 2014. Application of finite mixture of regression model with varying mixing probabilities to estimation of urban arterial travel times. *Transportation Research Record* 2442, 1 (2014), 96–105.
[5] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, San Francisco, 785–794.
[6] Daniel Delling, Andrew V Goldberg, Thomas Pajor, and Renato F Werneck. 2017. Customizable route planning in road networks. *Transportation Science* 51, 2 (2017), 566–591.
[7] Julian Dibbelt, Ben Strasser, and Dorothea Wagner. 2016. Customizable Contraction Hierarchies. *Journal of Experimental Algorithmics* 21, 1 (2016), 1–49.
[8] Karen K Dixon, Chi-Hung Wu, Wayne Sarasua, and Janice Daniel. 1999. Estimating free-flow speeds for rural multilane highways. *Transportation research record* 1678, 1 (1999), 73–82.
[9] Robert Geisberger, Peter Sanders, Dominik Schultes, and Daniel Delling. 2008. Contraction Hierarchies: Faster and Simpler Hierarchical Routing in Road Networks. *DBLP* 5038 (2008), 319–333.
[10] Chong Yang Goh, Justin Dauwels, Nikola Mitrovic, Muhammad Tayyab Asif, Ali Oran, and Patrick Jaillet. 2012. Online map-matching based on hidden markov model for real-time traffic sensing applications. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*. IEEE, IEEE, Anchorage,Alaska,USA, 776–781.
[11] Joshua S Greenfeld. 2002. Matching GPS observations to locations on a digital map. In *Transportation Research Board 81st Annual Meeting*, Vol. 22. ITF, Washington, DC, 576–582.
[12] Gang Hu, Jie Shao, Fenglin Liu, Yuan Wang, and Heng Tao Shen. 2016. If-matching: Towards accurate map-matching with information fusion. *IEEE Transactions on Knowledge and Data Engineering* 29, 1 (2016), 114–127.
[13] Michael P Hunter, Seung Kook Wu, and Hoe Kyoung Kim. 2006. Practical procedure to collect arterial travel time data using GPS-Instrumented test vehicles. *Transportation research record* 1978, 1 (2006), 160–168.
[14] MR Hustim and M Isran. 2013. The vehicle speed distribution on heterogeneous traffic: Space mean speed analysis of light vehicles and motorcycles in makassar-indonesia. In *Proceedings of the Eastern Asia Society for Transportation Studies*, Vol. 9. EASTS, Taipei, 599–610.
[15] Yin Lou, Chengyang Zhang, Yu Zheng, Xing Xie, Wei Wang, and Yan Huang. 2009. Map-matching for low-sampling-rate GPS trajectories. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*. ACM, Seattle Washington, 352–361.
[16] David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*. Association for Computational Linguistics, New York City, 152–159.
[17] Reham Mohamed, Heba Aly, and Moustafa Youssef. 2016. Accurate real-time map matching for challenging environments. *IEEE Transactions on Intelligent Transportation Systems* 18, 4 (2016), 847–857.
[18] Radford M Neal. 2000. Markov chain sampling methods for Dirichlet process mixture models. *Journal of computational and graphical statistics* 9, 2 (2000), 249–265.
[19] Paul Newson and John Krumm. 2009. Hidden Markov map matching through noise and sparseness. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*. ACM, Seattle Washington, 336–343.
[20] Takayuki Osogami and Rudy Raymond. 2013. Map matching with inverse reinforcement learning. In *Twenty-Third International Joint Conference on Artificial Intelligence*. AAAI Press / International Joint Conferences on Artificial Intelligence, Beijing, China, 2547–2553.
[21] Changxuan Pan, Jiangang Lu, Dawei Wang, and Bin Ran. 2007. Data collection based on global positioning system for travel time and delay for arterial roadway network. *Transportation research record* 2024, 1 (2007), 35–43.
[22] Byung-Jung Park, Yunlong Zhang, and Dominique Lord. 2010. Bayesian mixture modeling approach to account for heterogeneity in speed data. *Transportation research part B: methodological* 44, 5 (2010), 662–673.
[23] Sangjun Park, Hesham Rakha, and Feng Guo. 2011. Multi-state travel time reliability model: Impact of incidents on travel time reliability. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, IEEE, Washington, DC, USA, 2106–2111.
[24] Pu and Wenjing. 2012. Analytic Relationships Between Travel Time Reliability Measures. *Transportation Research Record Journal of the Transportation Research Board* 2254 (2012), 122–130.
[25] Henry Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory* 11, 3 (1965), 363–371.
[26] Shun Taguchi, Satoshi Koide, and Takayoshi Yoshimura. 2018. Online map matching with route prediction. *IEEE Transactions on Intelligent Transportation Systems* 20, 1 (2018), 338–347.
[27] Xiaodong Yu. 2009. Gibbs sampling methods for dirichlet process mixture model: Technical details. *University of Maryland, College Park* (2009), 1–18.
[28] Jing Yuan, Yu Zheng, Chengyang Zhang, Xing Xie, and Guang-Zhong Sun. 2010. An interactive-voting based map matching algorithm. In *2010 Eleventh international conference on mobile data management*. IEEE, IEEE, Kansas City, MO, USA, 43–52.
[29] Kai Zhao, Jie Feng, Zhao Xu, Tong Xia, Lin Chen, Funing Sun, Diansheng Guo, Depeng Jin, and Yong Li. 2019. DeepMM: Deep learning based map matching with data augmentation. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, Chicago, Illinois, USA, 452–455.
[30] Zhi-Hua Zhou and Ming Li. 2007. Semisupervised regression with cotraining-style algorithms. *IEEE Transactions on Knowledge and Data Engineering* 19, 11 (2007), 1479–1493.