Predicting Taxi and Uber Demand in Cities: Approaching the Limit of Predictability

Kai Zhao, Denis Khryashchev, Huy Vo

Abstract—Time series prediction has wide applications ranging from stock price prediction, product demand estimation to economic forecasting. In this paper, we treat the taxi and Uber demand in each location as a time series, and reduce the taxi and Uber demand prediction problem to a time series prediction problem. We answer two key questions in this area. First, time series have different temporal regularity. Some are easy to be predicted and others are not. Given a predictive algorithm such as LSTM (deep learning) or ARIMA (time series), what is the maximum prediction accuracy that it can reach if it captures all the temporal patterns of that time series? Second, given the maximum predictability, which algorithm could approach the upper bound in terms of prediction accuracy? To answer these two question, we use temporal-correlated entropy to measure the time series regularity and obtain the maximum predictability. Testing with 14 million data samples, we find that the deep learning algorithm is not always the best algorithm for prediction. When the time series has a high predictability a simple Markov prediction algorithm (training time 0.5s) could outperform a deep learning algorithm (training time 6 hours). The predictability can help determine which predictor to use in terms of the accuracy and computational costs. We also find that the Uber demand is easier to be predicted compared the taxi demand due to different cruising strategies as the former is demand driven with higher temporal regularity.

Index Terms—sharing economy; deep learning; predictive algorithm; predictability of time-series; data mining

1 Introduction

Traditional taxi systems in cities often suffer from inefficiencies due to uncoordinated actions as customer demand changes [1]. In some areas passengers experience long waits for a taxi, while in others, many taxis roam without riders. This leads to profit loss for taxi drivers, since vehicles are vacant when there is demand. Moreover, it reduces the level of passenger satisfaction because of the long wait times. The ability to predict taxi and Uber demand can help address the taxi-service inefficiency problem. With the deployment of the networked sensors and the widely used mobile phones in taxis, large amounts of information including location, time, number of passengers, weather, traffic, etc. can be collected in real time. This information provides opportunities to build an intelligent transportation system that is able to control and coordinate taxis at large scale and bring benefits to both taxi drivers and companies: taxi drivers can drive to high taxi demand areas, and ride-sharing companies (e.g. Uber) may re-allocate their vehicles using the surge pricing in advance to meet the passenger demand.

Recent studies have shown that the passenger demand information can be used in the taxi dispatch system to reduce passenger waiting time, taxi cruising time, or supply re-balancing cost [2], [3], [4]. In this paper we only study the taxi and Uber demand prediction problem. The design of the taxi dispatch system has been widely studied in the transportation and the operation research area and is not considered here. E.g., it has been found that with the taxi demand prediction component, the taxi dispatch system can reduce the average total taxi idle distance by 52%, and the supply demand error by 45% [5].

We define the taxi demand prediction problem as follows: given historical taxi demand data in a region, we want to *predict the number of taxis that will emerge* within the next time interval. Inspired by previous works [6], [2], [7], [8], [9], [10], we aim to predict the *met taxi demand*. We use the number of pick-ups as a representation of the taxi demand in a region, and treat them as time series data (see Fig. 1). Our method is general and can also be applied to predict the unmet taxi demand. As we discuss in Section 7.2, unmet demand can be inferred from the met taxi demand [11], [12]. In a recent report by the Taxi and Limousine Commission (TLC) [13], the agency that is responsible for regulating for-hire vehicles in New York City (NYC), there is a strong correlation of the socioeconomic impact to the taxi demand at various governing zones such as building blocks. As such, we elect to study our technique at the building block level.

Many methods have been proposed to predict taxi demand, including uncertainty analysis [6], probabilistic models [9], time series (ARIMA) [7], [8], SVM [2], and deep learning (LSTM) [14]. However, to apply these methods, we must answer two key questions:

- First, given a predictive algorithm such as LSTM or ARIMA, what is the maximum prediction accuracy that it can reach if it captures all the temporal patterns of that taxi/Uber demand time series?
- Second, given the maximum predictability, which algorithm could approach the upper bound in terms of prediction accuracy?

- *Kai Zhao is with the Robinson College of Business, Georgia State University, Atlanta, USA. E-mail: kaizhaofrank@gmail.com*
- *Denis Khryashchev is with the Graduate Center of the City University of New York, New York, USA. E-mail: dkhryashchev@gradcenter.cuny.edu*
- *Huy Vo is with the City College of the City University of New York, and the Center for Urban Science and Progress, New York University, New York, USA. E-mail: hvo@cs.ccny.cuny.edu*

In this paper, we answer these questions by analyzing the *maximum predictability* (Π^{max}) of taxi demand in a region to select the best predictor. The maximum predictability is defined by the *entropy* of the taxi demand time series, considering both the *randomness* and the *temporal correlation*. Maximum predictability Π^{max} was first introduced by Song et. al for analyzing user mobility [15]. Here, we define the maximum predictability Π^{max} as the highest potential accuracy of a taxi demand forecast that any predictive algorithm can reach.

The maximum predictability captures the degree of temporal correlation of the taxi demand time series by measuring the regularity of human mobility. For most regions, the taxi demand is governed by a certain amount of randomness (e.g. irregular events, basketball match) and some degree of regularity (e.g. weekly patterns, peak during 4-5pm weekdays), which can be exploited for prediction. For example, a building block with Π^{max} = 0.8 indicates that for about 20% of the time the taxi demand of this block appears to be totally random. In other words, no matter how good the predictive algorithm is, we cannot forecast the future taxi demand for a building block with $\Pi^{max} = 0.8$ with an accuracy that is higher than 80%. Π^{max} represents the fundamental limit for predictability of the taxi demand. The Π^{max} is a value between 0 and 1, the higher the more regular will the taxi demand be.

Previous work assumed that the maximum predictability (the degree of the temporal correlation) in different regions is the same, and proposed the use of a single predictive algorithm for all regions [7]. However, the strong temporal correlation of taxi demand does not always hold. Different regions have different functions and thus different predictabilities (see Section 5.2). Fig. 1 shows the hourly taxi demand for two building blocks in NYC. The taxi demand near the Metropolitan Museum of Art (MoMA) (Fig. 1 top) exhibits a strong temporal pattern. The regular peaks in MoMA happen during the weekends (especially after the closing time): people usually visit museums during weekends and leave after the closing time. In contrast, the taxi demand near the west port (Fig. 1 below) appears to be more random. There is no clear temporal pattern near the west port. This is because the taxi demand in a transportation hub such as the west port is heavily dependent on the arrival of ships and there is a high variability in their arrival times [16]. In fact, the taxi demand near MoMA has one of the highest predictabilities among all the building blocks in NYC, and the taxi demand near the west port has one of the lowest. Intuitively we should use different predictors forecasting the taxi demand in these two building blocks. For MoMA it is better to use a predictor that is able to capture the temporal correlation, for example, a Markov predictor. For the west port, a predictor that uses machine learning and can capture exogenous features such as the ship schedule may be more effective. We posit that to select the best predictor, we must analyze the *maximum predictability (Π^{max}) of the taxi demand in each region*.

In this paper we make three key contributions:

- We measure the theoretical maximum predictability[1] of the taxi demand for each building block in NYC. This represents the upper bound of the potential accuracy that a predictive algorithm α can reach. We show that the maximum predictability of the taxi demand can reach up to 83% on average capturing the degree of the temporal correlation of a taxi demand time series. Our findings indicate that taxi demand in NYC incorporates strong temporal patterns. We also find that the Uber demand is easier to be predicted compared the taxi demand due to different cruising strategies as the former is demand driven with higher temporal regularity.
- We implement and compare the prediction accuracy of five commonly used and representative predictors and examine their performance under different maximum predictability: the Markov (probability-based) [17], the Lempel-Ziv-Welch (LZW) (sequence modeling) [18], the auto-regressive integrated moving average (ARIMA) model (time series forecasting) [19], the Neural Network (NN) (machine learning) [10], and the Long Short-Term Memory (LSTM) (deep learning) [20]. Our results indicate that the maximum predictability is an approachable target for the actual prediction accuracy.
- We observe that the LSTM predictors provides better accuracy for building blocks with low predictability ($\Pi^{max} < 0.83$) by capturing hidden long-term temporal dependency, while the Markov predictor provides better accuracy for building blocks with high predictability ($\Pi^{max} \geqslant 0.83$). A compute-intensive deep learning predictor does not always outperform a simpler Markov predictor, while the latter requires only 0.02% computation time. Knowledge of the predictability can help determine which predictor to use in terms of the accuracy and computational costs. Using a third time-series data set, we show that our finding is general and can be applied to other time-series data sets.

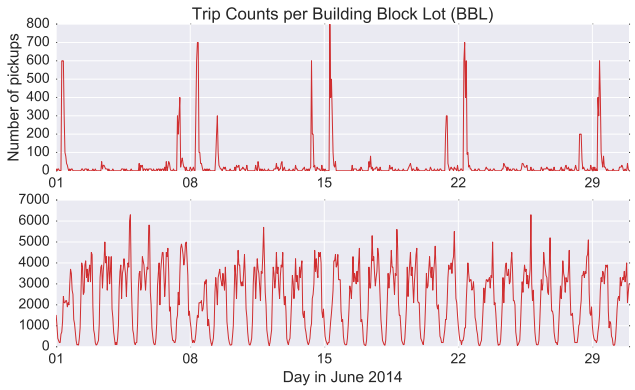


Fig. 1. Heterogeneous taxi demand time series in two building blocks: (top) a dense traffic area around the Metropolitan Museum of Art; and (bottom) a more open traffic area near the west port in NYC.

The remainder of this paper is organized as follows. Section II presents how we obtain the maximum predictability Π^{max}. Section III describes the five approaches we implemented for predicting taxi demand. In Section IV we

1. The code for calculating the maximum predictability can be found here: https://github.com/bdilab/Predictability

introduce the data sets we use in this paper and how we preprocess these data. We analyse the taxi demand predictability in Section V. In Section VI we compare the performance of the five predictive algorithms and show that predictability provides an effective measure for selecting appropriate prediction methods. Related work is addressed in Section VII. In Section VIII we discuss the impact of the spatial resolution on the prediction accuracy and generality of our findings. We conclude with Section VIII.

2 Predictability of Taxi Demand

In this section we formally define maximum predictability Π^{max}. Our goal is to solve the following problem:

Problem 1. *Given any predictive algorithm α and a sequence of taxi demand $D_n^{(i)}$ for time steps $1, 2, \dots n$ at a building block i, considering both the randomness and the temporal correlation of the taxi demand, find the maximum predictability (the highest potential accuracy) Π^{max} that a predictive algorithm α can reach.*

2.1 Taxi Demand

We use the number of taxi pick-ups $d_t^{(i)}$ to represent taxi demand at building block i at time step t ($1 \leq i \leq m$ and $1 \leq t \leq n$). For a given building block i, we have a sequence of the historical taxi demand $D_n^{(i)} = d_1^{(i)} d_2^{(i)} d_3^{(i)} \dots d_n^{(i)}$. Note that here $D_n^{(i)}$ represents taxi demand for time steps $1 \dots n$. For example, $D_n^{(i)} = 2, 1, 2, 2$ indicates that for block i, at time step 1 there were 2 pickups, at time step 2 there were 1 pickup, and so on. Table 1 summarizes the mathematical notation and main symbols used in the paper.

We only consider the total hourly pick-ups at each building block as time series and our goal is to predict the next hour pick-ups. We did not consider the problem of location recommendation in our paper.

Due to the high variability in taxi demand, it is hard to predict the exact value for $d_t^{(i)}$. To improve the performance of sequential predictors like Markov chain or LZW, we bin every value of the taxi demand in the corresponding range: $pq \leq d_t^{(i)} < (p+1)q$ where p and q are some natural numbers. We replace the original value of $d_t^{(i)}$ with the value pq and predict the rounded taxi demand. For example, if $q = 10$ and observed taxi demand $d_t^{(i)} = 621$, we assign it a new value of 620, for it is in the range $620 \leq d_t^{(i)} < 630$. After a few tests, we found $q = 10$ to make the prediction more relaxed while keeping the errors low. The effect of q on Π^{max} is shown in Appendix E.

2.2 Entropy

Entropy is an effective measure to characterize the degree of predictability. In general, low entropy corresponds to the higher predictability. We use three measures of entropy: the random entropy $S_{random}^{(i)}$, the Shannon entropy $S_{Shannon}^{(i)}$, and the real entropy $S_{real}^{(i)}$ [21].

2.2.1 Random Entropy

$$S_{random}^{(i)} = log_2 N^{(i)} \tag{1}$$

Here $N^{(i)}$ is the number of unique values in the taxi demand time series $D_n^{(i)}$, e.g. for $D_4^{(1)} = 2, 1, 2, 2$, we have $N^{(1)} = 2$. The lower number of unique values $N^{(i)}$ translates into the lower random entropy $S_{random}^{(i)}$ meaning that there are more repetitions in the sequence $D_n^{(i)}$ making it more predictable. However, random entropy does not take temporal order in $D_n^{(i)}$ into account.

Symbol	Meaning
$d_t^{(i)}$	Taxi demand at the building block i at time t.
$X_t^{(i)}$	Taxi demand as a random variable.
$D_n^{(i)}$	The historical taxi demand time series, $d_1^{(i)} d_2^{(i)} \dots d_n^{(i)}$.
$N^{(i)}$	Number of distinct values of the taxi demand at the building block i.
$S_n^{(i)}$	Time-ordered subsequence $S_n^{(i)}$, $S_n^{(i)} \subset D_n^{(i)}$.
$S_t'^{(i)}$	Length of the shortest unseen subsequence starting at time t.
$S_{random}^{(i)}$	Random entropy of $D_n^{(i)}$.
$S_{Shannon}^{(i)}$	Shannon entropy of $D_n^{(i)}$.
$S_{real}^{(i)}$	Real entropy of $D_n^{(i)}$.
Π	Predictability of an arbitrary predictive algorithm α.
Π^{max}	Upper bound of predictability Π, $\Pi \leq \Pi^{max}$.

TABLE 1
Mathematical notation and symbols used throughout the paper

2.2.2 Shannon Entropy

$$S_{Shannon}^{(i)} = -\sum_{j=1}^{N^{(i)}} p(d_j^{(i)}) log_2 p(d_j^{(i)}) \tag{2}$$

Here $p(d_j^{(i)})$ is the probability that the taxi demand at the building block i is equal to $d_j^{(i)}$, j represents the indices in the set of all unique values of taxi demand in $D_n^{(i)}$. Note that similar to the random entropy, Shannon entropy does not consider temporal patterns. For example, given two sequences of taxi demand $D_4^{(1)} = 2, 1, 2, 2$ and $D_4^{(2)} = 1, 2, 2, 2$, we have the values of random and Shannon entropy $S_{random}^{(1)} = S_{random}^{(2)}$ and $S_{Shannon}^{(1)} = S_{Shannon}^{(2)}$.

2.2.3 Real Entropy

$$S_{real}^{(i)} = -\sum_{S_n^{(i)} \subset D_n^{(i)}} P(S_n^{(i)}) log_2 [P(S_n^{(i)})] \tag{3}$$

$P(S_n^{(i)})$ represents the probability of finding a particular time-ordered subsequence $S_n^{(i)}$ in the taxi demand time series $D_n^{(i)}$. Unlike Shannon entropy and random entropy, the real entropy considers both the probabilities of the values in taxi demand time series and their temporal order [21].

The problem of finding all the subsets of a given set has exponential computational complexity ($O(2^n)$). We use the approximation proposed by Lempel and Ziv [22] that rapidly converges to the value of the real entropy. For a taxi demand time series of size n, the Lempel-Ziv approximation of real entropy is defined as

$$S_{real}^{(i)} \approx \left(\frac{1}{n}\sum_{t} s_t'^{(i)}\right)^{-1} \ln n \quad (4)$$

Here, $s_t'^{(i)}$ represents the length of the shortest subsequence that starts at the time t and does not appear from time period 1 to $t-1$.

To illustrate that real entropy $S_{real}^{(i)}$ captures temporal regularity in sequences we compare two sequences of taxi demand $D_{10}^{(1)}$ of building block 1:

$$10, 20, 10, 20, 10, 20, 10, 20, 10, 20$$

and $D_{10}^{(2)}$ of building block 2:

$$20, 10, 10, 20, 20, 20, 10, 10, 20, 10.$$

Both sequences contain only 2 unique values: 10 and 20. Each value appears with a probability of 50% leading to the same mean value of 15 and a standard deviation of 5 for both sequences. Their values of the Shannon and the random entropy are equal as well:

$$S_{random}^{(1)} = S_{random}^{(2)} = S_{Shannon}^{(1)} = S_{Shannon}^{(2)} = 1.0$$

Yet, time series $D_{10}^{(1)}$ is quite regular, knowing that the current value is 10 allows one to predict with certainty that the next values will be 20. Time series $D_{10}^{(2)}$ is more unpredictable and does not exhibit a temporal pattern that could be put to use for its forecasting.

The real entropy is able to quantify the difference in terms of temporal regularity between these two sequences. The the real entropy for $D_{10}^{(1)}$ is 0.82 while for $D_{10}^{(2)}$ it is 1.00, indicating a lower real entropy and a higher temporal regularity for the taxi demand time series $D_{10}^{(1)}$, which is easier to be predicted.

2.3 Maximum Predictability Π^{max}

We define the predictability Π as the success rate of correct predictions of future taxi demand made with the algorithm α. The predictability measure Π associated with every building block i is subject to the Fano's inequality, $\Pi \leq \Pi^{max}$ [15]. Given the value of entropy S and the number $N^{(i)}$ of distinct values in taxi demand time series at a building block i, the maximum predictability Π^{max} is defined by the following equation:

$$S = -\Pi^{max} log_2(\Pi^{max}) - (1-\Pi^{max}) log_2(1-\Pi^{max}) + (1-\Pi^{max}) log_2(N^{(i)}-1) \quad (5)$$

If a location has entropy $S = 0$, then this location's taxi demand is completely regular and thus the it is fully predictable. If, however, a location entropy $S = S_{random} = log_2 N$, then its taxi demand is expected to follow a totally random pattern and thus we cannot forecast it with accuracy that exceeds $1/N$. Most locations have a finite entropy laying between 0 and S_{random}, indicating not only that a certain amount of random events governs their future taxi demand, but also that there is some regularity in their taxi demand that can be exploited for predictive purposes.

The maximum predictability Π^{max} takes values in range $[0, 1]$. The larger the value, the bigger forecasting accuracy can be achieved with the algorithm α. We quantify the limits of predictability of a location's taxi demand based on its taxi demand history. We answered the following question: how predictable is a locationss taxi demand at next time interval given the entropy of its history? We use a version of Fanos inequality to relate the upper bound of predictability to the entropy of a locations past history of taxi demand. A full proof of the predictability upper bound Π^{max}, which is inspired by the proof from Song et al. [15], is provided in the Appendix.

With different entropy measures, $S_{random}^{(i)}$, $S_{Shannon}^{(i)}$ and $S_{real}^{(i)}$, we have different measures of maximum predictability: Π^{random}, $\Pi^{Shannon}$ and Π^{real}. Since $S_{random}^{(i)} \geq S_{Shannon}^{(i)} \geq S_{real}^{(i)}$, it has been proven that $\Pi^{random} \leq \Pi^{Shannon} \leq \Pi^{real}$ [15], making Π^{real} the most accurate approximation of maximum predictability Π^{max}. Thus Π^{real} is Π^{max} in our paper. Now, calculating the value of Π^{real}, we can answer the problem proposed in the beginning of the section: given a predictive algorithm α, find the maximum predictability Π^{max} that the predictive algorithm α can reach. The detailed analysis of the maximum predictability of each building blocks in NYC can be found in Section 5.

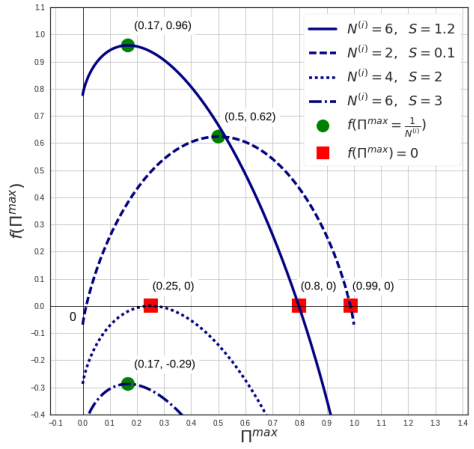


Fig. 2. Property of the maximum predictability. Three cases with and one case without solutions: $N^{(i)} > 2^S$, $S \to 0, N^{(i)} = 2^S$, $N^{(i)} < 2^S$.

2.4 Scalability

In this section, we provide a scalable algorithm (see Appendix Alg. 1) for calculating the maximum predictability Π^{max} using the properties derived in Appendix D. Currently Matlab solver is the only existing method we can find to calculate the maximum predictability [15]. We find that it is extremely slow and not scalable. We examine the properties of the maximum predictability and provide an efficient algorithm to calculate the maximum predictability.

The maximum predictability Π^{max} can be obtained by solving Equation 5. We move S to the right side of the equation, and we define the function

$$f(\Pi^{max}) = -\Pi^{max} log_2(\Pi^{max}) - (1-\Pi^{max}) log_2(1-\Pi^{max}) + (1-\Pi^{max}) log_2(N^{(i)}-1) - S \quad (6)$$

Since Π^{max} is a value between 0 and 1, finding the intersection when $f(\Pi^{max}) = 0$ can solve the equation

with $\Pi^{max} = \gamma$ given the values of $N^{(i)}$ and S. We use Householder's method [23] and apply 4 lemmas (proven in the Appendix D) to decrease the overall computation time (see Appendix Alg. 1):

LEMMA 2. *$f(\Pi^{max}) = 0$ has no solutions if $N^{(i)} < 2^S$.*

LEMMA 3. *$f(\Pi^{max}) = 0$ has exactly one solution if $N^{(i)} = 2^S$ and the solution is $\frac{1}{N^{(i)}}$.*

LEMMA 4. *$f(\Pi^{max}) = 0$ has at most two solutions if $N^{(i)} > 2^S$ and the biggest solution is in $\left(\frac{1}{N^{(i)}}, 1\right)$.*

LEMMA 5. *If $S \to 0$, the solution to $f(\Pi^{max}) = 0$ approaches 1.*

Fig. 2 shows the shapes that the function $f(\Pi^{max})$ takes in accordance with the lemmas. Since both $S^{(i)}$ and $N^{(i)}$ are known numbers for a building block i, the computation time of solving all the equation $f(\Pi^{max}) = 0$ is $O(m)$, where m is the total number of building blocks. We can scale by distributing the computation of $f(\Pi^{max}) = 0$ for each building block over multiple processors.

To test the performance of our scalable algorithm, we implement a Matlab solver (R2016a, The MathWorks) for solving the same maximum predictability equation. To the best of our knowledge, the solver uses Dekker's algorithm [24] which should converge within $log_2^2((b-a)/\delta)$ [25] function evaluations for the values of its argument in $[a, b]$ and error tolerance δ. In our case it is $log_2^2(1/\delta)$ for we have $0 \leq \Pi^{max} \leq 1$. For example, if we set $\delta = 10^{-8}$, it will require 700+ function evaluations to find the zeros of (6).

Our algorithm relies on the Householder's method of order 2, and has a cubic rate of convergence in the worst case. Our initial guess is $\Pi^{max} = (N^{(i)} + 1)/(2N^{(i)})$ based on lemma 4 (see line 19 of Appendix Alg. 1) allows us to find the solution even faster. In special cases (lemmas 1-5) we can find the solution in constant time. Given the accuracy parameter, the worst-case number of iterations is log log (1/accuracy). For accuracy of up to 10^{-16}, the algorithm converges within 10 iterations. Overall, in practical applications it takes less than 30 function evaluations to calculate the value of Π^{max} with Appendix Alg. 1.

Experimentally we observe that our algorithm is 2 orders of magnitude faster (about 200 times) than the Matlab solver while solving the same tasks (seconds vs hours), which verifies the efficiency of our algorithm.

3 PREDICTORS

We employ five predictors: the Markov chain (probability-based) [17], the LZW (sequence modeling) [18], ARIMA [19] (time series forecasting), Neural Network (machine learning) [10], and LSTM (deep recurrent neural network) [20] for predicting the taxi demand, and compare their performances given the value of the maximum predictability Π^{max} that quantifies temporal regularity of the series.

In this section, we answer the following question:

Problem 2. *Given the maximum predictability Π^{max} for a building block i, we want to find the predictor that is more efficient in terms of the trade-off between its computation time and forecasting accuracy.*

3.1 Markov chain Predictor

We propose the order-k $O(k)$ Markov chain predictor [26] to forecast the future taxi demand from the subsequence of k most recent observed values $d_{n-k+1}^{(i)}, d_{n-k+2}^{(i)}, \ldots, d_n^{(i)}$. The key idea of Markov predictor is to find the value β that has the highest probability to follow the sequence of the last k values observed in the time series. For example, consider taxi demand series $\{1,1,2,2,0,1,1,2,2,3,1,1,2,2,0,1,1,2,2\}$ that contains a repeating subsequence $\{1,1,2,2,0\}$ and a noisy value at 10^{th} position (3 instead of expected 0). Markov chain predictor of order 5 will look for historical occurrences of subsequence $\{1,1,2,2\}$ and will evaluate two probabilities: $P(0|\{1,1,2,2\}) = 2/3$ and $P(3|\{1,1,2,2\}) = 1/3$. Clearly, 0 following $\{1,1,2,2\}$ has a higher probability and it will be selected as a future value of taxi demand $\hat{\beta}$.

More formally, we denote the taxi demand during time period t at the building block i as $X_t^{(i)}$. Let $X_{(t,t+k)}^{(i)}$ denote the sequences of taxi demand that start at time t and end at time $t+k$: $X_t^{(i)}, X_{t+1}^{(i)}, X_{t+2}^{(i)}, \ldots, X_{t+k}^{(i)}$ for $1 \leq t \leq n-k$. Considering the location i that has a taxi demand history $D_n^{(i)} = d_1^{(i)} d_2^{(i)} d_3^{(i)} \ldots d_n^{(i)}$ with $N^{(i)}$ unique values as shown in Section 2.1, and following Markov assumption we have

$$\begin{aligned} P\Big(X_{n+1}^{(i)} = \beta | X_n^{(i)} = D_n^{(i)}\Big) \\ = P\Big(X_{n+1}^{(i)} = \beta | X_{n-k+1,n}^{(i)} = c\Big) \\ = P\Big(X_{t+k+1}^{(i)} = \beta | X_{t+1,t+k}^{(i)} = c\Big) \end{aligned}$$

Here $P\left(X_{n+1}^{(i)} = \beta | X_n^{(i)} = D_n^{(i)}\right)$ means that there is a demand for β taxis at the building block i during the time interval $n+1$. c is a repeating subsequence of taxi demand within $D_n^{(i)}$, i.e. $d_{n-k+1}^{(i)} d_{n-k+2}^{(i)} \ldots d_n^{(i)} = d_{t+1}^{(i)} d_{t+2}^{(i)} \ldots d_{t+k}^{(i)} = c$. We propose a transition probability matrix T as the Markov chain predictor (see Appendix Alg. 5) with its rows and columns representing the taxi demand time seriess of length k. Each element $T_{c,c'}$ in the matrix represents the prediction $T_{c,c'} = P\left(X_{n+1}^{(i)} = \beta | X_n^{(i)} = D_n^{(i)}\right)$, where $c = d_{n-k+1}^{(i)} d_{n-k+2}^{(i)} \ldots d_n^{(i)}$ and $c' = d_{n-k+2}^{(i)} d_{n-k+3}^{(i)} \ldots d_n^{(i)} \beta$. The matrix T provides the immediate probability $P\left(X_{n+1}^{(i)} = \beta | X_n^{(i)} = D_n^{(i)}\right)$ of the future taxi demand of β following the observed sequence $D_n^{(i)}$.

As T is unknown, we define an estimate probability P' from the taxi demand history $D_n^{(i)}$ as

$$P'\Big(X_{n+1}^{(i)} = \beta | X_n^{(i)} = D_n^{(i)}\Big) = \frac{C\Big(c\beta, D_n^{(i)}\Big)}{C\Big(c, D_n^{(i)}\Big)} \qquad (7)$$

Here $\frac{C\left(c\beta, D_n^{(i)}\right)}{C\left(c, D_n^{(i)}\right)}$ is the probability of observing the subsequence c followed by β in the observed taxi demand $D_n^{(i)}$, $C\left(c, D_n^{(i)}\right)$ stands for the number of time subsequence c occurred in $D_n^{(i)}$.

The formula allows us to estimate the future taxi demand at the the building block i given the order-k $O(k)$ Markov chain predictor matrix T. In Appendix Alg. 5 we choose the β with the highest estimated probability P' given the

matrix T. Note that the Markov chain predictor might return an empty value given the recent taxi demand time series c. This is due to the fact that in the observed taxi demand history subsequence $d^{(i)}_{n-k+1} d^{(i)}_{n-k+2} \dots d^{(i)}_n = c$ did not occur. In this case, we return the taxi demand with highest probability in $D^{(i)}_n$. We set $k = 3$ here to improve the prediction accuracy while reducing the computation time (see Section 6.2).

It is easy to maintain and update the Markov chain predictor matrix T in Appendix Alg. 5. The predictor scans the matrix T at row c and picks the next taxi demand with the highest probability as its forecast. Then it updates the row with the forecast. Note that here T is a $c' \times N^{(i)}$ matrix, c' is the number of unique subsequences in $D^{(i)}_n$.

3.2 Lempel-Ziv-Welch Predictor

The LZW predictor is based on the Lempel-Ziv-Welch text encoding algorithm (LZW algorithm). The predictor relies on building a tree of the shortest subsequences that have not been observed by time t. Then the prediction is made based on the probability of observing a value β following the shortest subsequences. For example, for taxi demand series $\{1, 1, 2, 1, 1, 2, 1, 3, 2\}$ LZW will evaluate probability of a value following the shortest subsequence matching the most recent values of the series: 2 being the shortest subsequence at $t = 3$ was followed by 1 after $\{1, 1, 2\}$, i.e. 1 will be the predicted value with probability 1.

In detail, given a taxi demand time series $D^{(i)}_n$, LZW algorithm partitions $D^{(i)}_n$ into a set of distinct subsequences $\left\{s^{(i)}_0, s^{(i)}_1, s^{(i)}_2, s^{(i)}_3, \dots s^{(i)}_m\right\}$, where $s^{(i)}_t$ represents the shortest subsequence that starts at the time interval t and does not appear up until $t-1$ (see Appendix Alg. 2). For example, in Figure S1 (see Appendix) the taxi demand history $D^{(i)}_n = \{1, 1, 2, 1, 1, 2, 1, 3, 2\}$ can be parsed into the set $\{1, 2, 3, 11, 12, 21, 121, 13, 32\}$ by LZW algorithm.

We build a LZW tree to maintain the LZW predictor (see Appendix Alg. 2). The tree grows dynamically parsing the taxi demand history $D^{(i)}_n$. The root of the tree is an empty list. All immediate children of the root node represent taxi demand subsequences $s^{(i)}_t$ of length 1, e.g. $1, 2, 3$. Each child node of the root combined with each of its immediate children nodes represent taxi demand subsequences $s^{(i)}_t$ of length 2, e.g. $11, 12, 13, 21, 32$, and so on. Every node also records information about the number of times the subsequence of its parents and itself $s^{(i)}_t$ occurred in $D^{(i)}$, e.g. 1 occurred 4 times, 2 occurred twice, 12 occurred twice and so on. Denoting the taxi demand at time period t at the building block i as a random variable $X^{(i)}_t$, we define the LZW predictor as:

$$P\left(X^{(i)}_{n+1} = \beta | D^{(i)}_n\right) = \frac{N_{LZ}\left(s^{(i)}_m \beta, D^{(i)}_n\right)}{N_{LZ}\left(s^{(i)}_m, D^{(i)}_n\right)} \tag{8}$$

Similarly to Markov chain predictor, here $\frac{N_{LZ}\left(s^{(i)}_m \beta, D^{(i)}_n\right)}{N_{LZ}\left(s^{(i)}_m, D^{(i)}_n\right)}$ represents the probability that the subsequence $s^{(i)}_m \beta$ occurs in $D^{(i)}_n$ calculated as the number of times the subsequence $s^{(i)}_m$ preceded β divided by the total number of times it was observed. While the Markov chain predictor considers how often a subsequence of interest occurs in the entire taxi demand time series $D^{(i)}_n$, the LZW predictor only considers the occurrences of the shortest subsequences $s^{(i)}_t$ observed at time t.

3.3 Neural Network Predictor

The Neural Network (NN) predictor we employ (see Appendix Alg. 3) consists of two layers with sigmoid and softmax activation functions [27] (see Figure S2 in Appendix). We adopt the same exogenous input data, i.e., "temperature", "precipitation", "wind speed", "day of the week", and "hour of the day" that we used in our previous research for predicting taxi demand [10]. Each of the hidden layers contains K neurons with the output calculated as

$$S_j\left(\sum_k w^{(j)}_k x^{(i)}_k + b^{(j)}\right) \tag{9}$$

Here $j \in \{1, 2\}$ enumerates the hidden layers, k iterates over the K neurons of a hidden layer, $w^{(j)}_i$ represents the layer's weights, $b^{(j)}$ stands for the layer's bias, S_j is an activation function, and $x^{(i)}_k$ denotes the input that is composed of the values of taxi demand $d^{(i)}_t$ combined with the exogenous variables. When the neural network is trained, the predicted value of future taxi demand β is linearly combined with m previous values of observed taxi demand $\{d^{(i)}_{n-m}, d_{n-m+1}, \dots, d^{(i)}_n\}$ using the Gaussian kernel

$$\beta' = \frac{1}{\sqrt{2\pi}\sigma}\left(\beta + \sum_{t=1}^{m} d^{(i)}_{n-t} e^{\frac{-(n-t)^2}{2\sigma^2}}\right) \tag{10}$$

where β' stands for the final prediction made by the Neural Network predictor.

One of the characteristics of Neural Network based predictors is the vast number of parameters and hyperparameters to optimize. Among the most important parameters are the number of layers and neurons per layer. Experimenting with the number of layers between 1 and 4 we did not acquire a significant decrease of the mean squared error having the number of layers bigger than 2, while the execution time increased significantly.

Having selected 2 hidden layers, we run experiments with the number of neurons per layer that varied between 10 and 100 in steps of 15. Mean sMAPE errors (See Section 6.1 Experiment Setup) given K neurons per layer are shown on Supplementary Fig S3. We found 55 neurons to be optimal. We used Yellow Taxi dataset to find the optimal number of neurons in the following manner:

1) Sort taxi demand series $D^{(i)}_n$ based on the value of maximum predictability Π^{max} and bin them into 10 bins: $0 \le \Pi^{max} < 0.1$, $0.1 \le \Pi^{max} < 0.2$, …, $0.9 \le \Pi^{max} \le 1$. Within every bin select 10 time series that have the average values of Π^{max};
2) Use the first 80% of the values of every selected taxi demand time series for the training set, and the last 20% for the test set. Train the Neural Network predictor with K neurons on the training set.

3) Select the number of neurons K that has the lowest mean sMAPE error on the test set.

3.4 ARIMA Predictor

The fourth predictor that we incorporate in our framework is the ARIMA model [19]. We use ARIMA as a representative of the time-series predictor here and test its performance under different levels of predictability. The model consists of three parts: auto-regression, integration, and moving average. For example, given taxi demand series $\{1, 1, 2, 2, 0, 1, 1, 2, 2, 3, 1, 1, 2, 2, 0, 1, 1, 2, 2\}$ we can fit a model consisting of auto-regression, integration, and moving average of order 1: $d_t^{(i)} = 0.016 + 1.14 \cdot d_{t-1}^{(i)}$ that will effectively predict $\hat{\beta} = 1.62$.

Order of the model can be denoted as ARIMA(p, d, q), where p is the number of auto-regressive terms, d is the number of nonseasonal differences (integration) that introduce nonstationarity when $d \neq 0$, and q is the number of error terms (moving-average). The details of the ARIMA model can be found in the Appendix A.1 ARIMA.

A standard way to identify the proper order of the model is to the Akaike Information Criterion (AIC) [28]. The optimal model ARIMA(p, d, q) is selected as the one that has the lowest AIC coefficient compared with the other orders considered. To improve the prediction accuracy and decrease the number of overall iterations, one might consider relating the choice of the parameters p and q with the inverse frequencies of largest magnitudes in the frequency spectrum of the original time series X. Alternative approaches to select the parameters (p, d, q) minimize $\sum_{i=1}^{N} \epsilon_i^2$.

To improve the prediction accuracy and decrease the number of overall iterations, we choose each of the parameters, p and q from the set $\{1, 2, 4, 8, 12, 24\}$ that represents the periods (1/frequencies) of the Fourier Transformation that have the highest magnitudes (see Section 6.2). The general logic of the ARIMA predictor that we use for taxi demand forecasting is shown in Appendix Algorithm 4.

3.5 LSTM Predictor

LSTM [20] is a Recurrent Neural Network designed specifically for modeling sequential and time series data as it can capture long-term dependency. It has been found that LSTM outperforms other commonly used predictors when predicting real-time taxi demand in cities [14]. To test how LSTM performs under different predictability comparing to other predictors we implement a LSTM predictor in our settings. We design our LSTM model as follows:

We used a standard LSTM model as shown in Supplementary Fig. S4. $D_n^{(i)}$ stand for the inputs, h_n is an output of the n^{th} cell, $\otimes$, $\oplus$, and $\tanh$ inside an ellipse represent pointwise multiplication, addition, and application of $\tanh$ correspondingly. σ and $\tanh$ inside rectangles denote standard neural network layers (neurons) with sigmoid and hyperbolic tangent activation functions.

The inputs for the model enter LSTM cells that contain 3 gates within their structure: input, output, and forget. The details of the LSTM model can be found in the Appendix A.2 LSTM. We optimize the number of layers and lags (how many time steps to account for) for our LSTM model. We sampled 10 series from the Yellow Taxi dataset with various maximum predictability values. We trained LSTM models with 2 to 4 layers and 2 to 10 lags for every of the selected time series. As our parameters we picked a model with 2 layers and a lag size of 2 for the model had the lowest mean prediction errors.

4 Data Sets and Preprocessing

In this section, we introduce the data sets we used in this paper and how we preprocess the data. We use the NYC yellow taxi data set, the NYC Uber taxi data set, and the NYC land use data set. These data sets provide a good coverage of the city with respect to both space and time.

4.1 Data Sets

	NYC Yellow Taxi	NYC Uber
Measurement	GPS	GPS
Number of samples	13,813,031	663,845
Time	June, 2014	June, 2014
Accuracy	3 m	3 m
Number of participants	13,237	Unknown

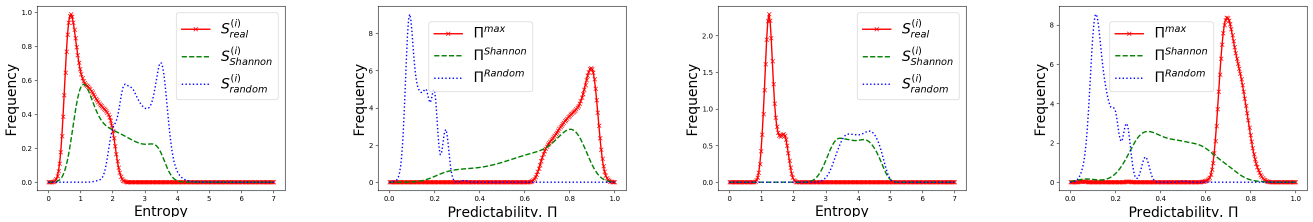
TABLE 2
The New York City Taxi and Uber Data sets

Taxi data. The NYC yellow taxi data set is a public data set provided by the Taxi and Limousine Commission (TLC) [2]. It includes trip records from all trips completed in yellow taxis in NYC in June, 2014. Each trip record contains pick-up and drop-off time, pick-up and drop-off locations, trip distances, itemized fares, and driver-reported passenger counts. The data was recorded through meters installed in each taxi. In total we have 13,813,031 taxi pick-up records from 13,237 yellow taxis.

The NYC Uber data set is a public data set provided by the TLC [3]. This data set contains 663,845 Uber pick-up records in June 2014, about 4.8% of the total yellow taxi pick-ups in NYC. The key information provided by these data sets is summarized in Table 2. We extract the following information from the data set: taxi id, pickup time and the corresponding pick-up location (building block).

Land use data. We use the NYC PLUTO (Primary Land Use Tax Lot Output) data set to map the yellow taxi GPS points to the associated building blocks and provide the land use information for each building block [29]. The PLUTO data set is an extensive land use and geographic data set provided by NYC Department of City Planning. It contains detailed information about every piece of land in the city, including year built, number of units, and lot size. Since the Uber data set is very sparse (only 4.8% of the total yellow taxi pick-ups), we use the NYC neighborhood data set [29], and map the Uber pick-up records with associated the neighborhoods instead of building blocks.

2. http://www.nyc.gov/html/tlc
3. https://github.com/fivethirtyeight/uber-tlc-foil-response



(a) Entropy (Hourly) (b) Predictability (Hourly) (c) Entropy (Daily) (d) Predictability (Daily)

Fig. 3. Distribution of (a) Hourly Entropy, (b) Hourly Predictability, (c) Daily Entropy, and (d) Daily Predictability of the taxi demand over all building blocks. The red line, the green line and the blue line refer to the probability density function of $S^{(i)}_{real}$, $S^{(i)}_{Shannon}$, $S^{(i)}_{random}$ and Π^{max}, $\Pi^{Shannon}$, Π^{random} in (a) (c) and (b) (d) respectively.

4.2 Data Preprocessing

We use the PLUTO shape file to map the yellow taxi pick-up GPS points to the associated building blocks: if a building block is within 200 ft radius of the pick-up location, we consider that building as the one passengers getting in the taxi and there is one taxi demand at that building block. If multiple building blocks are within 200 ft radius of the pick-up location, we consider the nearest building block as the one that has a taxi demand. There are over 43,000 building blocks in Manhattan, however, for meaningful analysis we have only considered pickups from the blocks that have sufficient temporal coverage: such blocks have at least 5 pickups a day each. Only 9940 blocks satisfied our criterion. The number of dropped pickup records is about 16% (2257296 removed records).

For the Uber data sets, we map matching the Uber pick-up location records with the neighborhood shape file. We found that there are 75 out of the 190 neighborhoods in NYC with least 5 Uber picks a day. After the map matching process, we obtain the taxi demand time series for each building block or the Uber demand time series for each neighborhood (demand time series $D^{(i)}_n = d^{(i)}_1 d^{(i)}_2 d^{(i)}_3 \dots d^{(i)}_n$).

Scalability. All of our data preprocessing were conducted using the operational data facility at our research center. In particular, the mapping of taxi pickups to geospatial features, which requires a lot of processing given the volume of the pickup trips, was done on a 1200+ core cluster running Cloudera Data Hub 5.4 with Apache Spark 1.6. The cluster consists of 20 high-end nodes, each with 24TB of disk, 256GB of RAM, and 64 AMD cores. It takes about ten minutes for our R-tree based [30] algorithm to map matching the 14 million samples. Other state-of-the-arts data indexing structures such as Elite [31] could also be used for efficient, parallel update and query processing by leveraging peer-to-peer and parallel computing techniques.

5 Results

In this section, we analyse the predictability of taxi demand over each building block in NYC. We show that the maximum predictability of the taxi demand can reach up to 83% accuracy in average, indicating a high temporal regularity of human mobility (daily/weekly patterns).

5.1 Limits of Predictability

We show the distribution of the entropy and the maximum predictability obtained from the yellow taxi data set in Fig. 3.

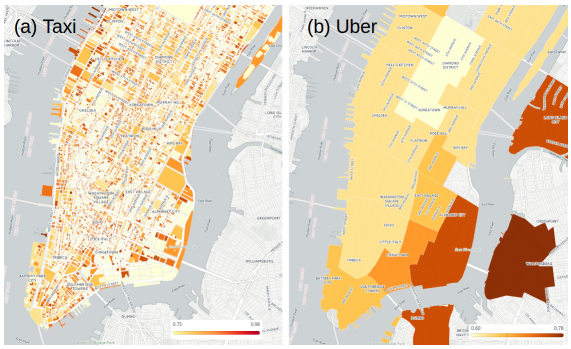


Fig. 4. Predictability in Manhattan, going from yellow (low) to red (high).

We first determine the entropy S and maximum predictability Π of the taxi demand of each building block using the yellow taxi data set. Then we obtain the distribution of the entropy S and maximum predictability Π over all building blocks. Fig 3 (a) depicts the entropy distribution of real Entropy $S^{(i)}_{random}$, Shannon Entropy $S^{(i)}_{Shannon}$ and random Entropy $S^{(i)}_{real}$; whereas Fig 3 (b) presents the distribution of Π^{max}, $\Pi^{Shannon}$, Π^{random} respectively.

We find that the distribution of $S^{(i)}_{random}$ peak at $S^{(i)}_{random} = 3.6$ (see Fig. 3 (a)). It implies that a building block would have $N^{(i)} = 2^{3.6} \approx 12$ distinct taxi demand levels. That is, almost every two hours we will observe a new taxi demand level compared to the previous one. Recall that we define $q = 10$ when categorizing the taxi demand, which implies that there are about 10 taxi demand differences every two hour in the same building block. In contrast, the real entropy $S^{(i)}_{real}$ peaks at a much smaller entropy value, $S^{(i)}_{real} = 0.9$. As discussed in Section. 2.2, the real entropy captures the temporal correlation of the taxi demand time series. The small real entropy $S^{(i)}_{real}$ means a high temporal pattern.

The predictability that any algorithm can correctly predict the next taxi demand is Π, and the upper bound is Π^{max}. From the distribution of the maximum predictability Π^{max} shown in Fig. 3 (b), we observe that average value of Π^{max} is 0.83. It indicates that the taxi demand can potentially be correctly predicted with an accuracy 83% over all building blocks. Both Π^{random} and $\Pi^{Shannon}$ are smaller than Π^{max}, which is constant in the previous findings [15], $\Pi^{random} \leq \Pi^{Shannon} \leq \Pi^{max}$. Since Π^{max} captures the temporal correlation of taxi demand, we can reach a higher potential predictive accuracy if considering the temporal

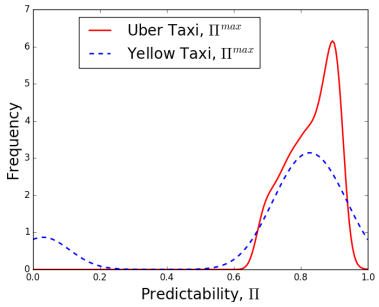


Fig. 5. Distribution of Π^{max} of the Uber and yellow taxis

correlation in our predictive algorithms. Similar results can also be found if we group the taxi demand daily instead of hourly (see Fig. 3 (c) and (d)). It means that there is a strong temporal pattern of human mobility at different temporal resolution. We apply a Fourier transformation over the taxi demand to further understand the periodicity in Section 6.2.

5.2 Predictability of Different Functional Buildings

In Fig. 4 (a) we plot the heat map of hourly taxi demand predictability in Manhattan. From the figure we observe that different building blocks exhibit different maximum predictability. In the working places such as Lower Manhattan or residential places such as East Village, they have higher predictability (temporal correlation). People usually go work in the morning and come home during night, thus the predictability in these areas are higher compared to other places. In other places such as Chelsea, NYC's premier contemporary-art district, or Little Italy, NYC's premier restaurant district, the temporal correlation is not as high as working or residential places.

To further evaluate the predictability of different functional regions, in Table 3 we show the predictability of taxi demand of building blocks with different land use. Both residential and working places have high predictability and thus exhibit strong temporal patterns. For other places such as hotels, transportation centers or bureau properties, the predictability is not as high as residential or working places. The taxi demand for these regions is mainly dependent on the events happened during the day (e.g. a boat arrives at the pier or guests check out of a hotel).

5.3 Uber Versus Yellow Taxi

In this section we examine the difference of the predictability between the Uber and yellow taxis. Due to the sparsity of the Uber taxis, we analyze the maximum predictability of Uber data sets at neighborhood level (see Fig. 4 (b) for the heatmap). We also group the hourly taxi demand in each neighborhood and examine the maximum predictability of the hourly taxi demand time series. As shown in Fig. 5, the maximum predictability of the Uber taxi service is higher than the yellow taxis. This is possibly because the yellow taxis usually use a random cruising strategy, while the Uber taxis go to the passenger's places when a request is received. The temporal correlation of the taxi demand in a region can be better captured by the Uber taxi.

6 Evaluation: Approaching the Limit of Predictability

In this section, we evaluate five predictors, the Markov [17], the LZW [18], the ARIMA [19], the NN [10] the LSTM [14] and examine which prediction algorithm can approach the prediction upper bound Π^{max}.

6.1 Experiment Setup

We conduct our experiments with five predictors on a workstation with dual Intel Xeon E5-2695 2.4GHz processors (12 cores in total), and 32 GB of memory. Our algorithms were implemented in Python with the use of NumPy, scikit-learn, and TensorFlow libraries. Inspired by previous work [7], we compare our predictors training them on the first three weeks of pickup data, and predicting the last week. We aggregate taxi pickups hourly for each building block, then rank the building blocks by the maximum predictability Π^{max}, and group them into ten 1,000 building block groups. Due to LSTM and NN predictors being computationally expensive, we sample time series from the set of median building blocks of each group to evaluate the predictors (10 in each group). To justify our sample size, we ran two sample KS test [32] first comparing original distributions of the sample time series against the rest, and then comparing the distributions after differencing time series with lags 1, 2, 3, 4, 8, 12, and 24 that were used to select parameters of our predictors (see Section 6.2). Our null hypothesis was that pickups in our sample time series share the same distribution with the rest of the series. Setting our $\alpha = 0.05$ we can reject only a small amount of samples as coming from a different distribution (see Supplementary Table SIII).

Cross-Validation In order to reduce the bias in our experiments and evaluate the true performance of our predictors, we used cross-validated training and testing schema. The training and testing process starts with training a model on the first 80% of the values of a time series and predicting the value that follows the training set to pick up the best parameters for the NN and LSTM predictors. In total we have used 14,400 time series.

Error metrics. We use symmetric Mean Absolute Percentage Error (sMAPE [33]) to evaluate the performance of our predictors. Denoting the forecasted taxi demand time series with $F^{(i)} = f_1^{(i)} f_2^{(i)} \dots f_n^{(i)}$ and the actual demand time series with $D_n^{(i)} = d_1^{(i)} d_2^{(i)} \dots d_n^{(i)}$ and noticing that both $d_j^{(i)}$ and $f_j^{(i)}$ are non-negative numbers we define the sMAPE metrics as follows:

$$sMAPE^{(i)} = \frac{1}{n} \sum_{t=1}^{n} \frac{\left| d_t^{(i)} - f_t^{(i)} \right|}{d_t^{(i)} + f_t^{(i)} + c} \tag{11}$$

Due to sMAPE metrics criticized [34] for being undefined for 0 values and having a skewed distribution for values that are close to 0, we add a constant to the denominator. Overall, sMAPE metrics allows us to compare the forecast errors with our previous work [35], and with the theoretical limit of predictability computing $1 - \Pi^{max}$. The prediction accuracy equals to $(1-$ prediction error$)$, where we use sMAPE to evaluate the prediction error in our paper.

Building Classes: First Level	Building Classes: Second Level	Maximum Predictability Π^{max}
Y. GOVERNMENT INSTALLATIONS	Y8. Department of Public Works	**0.92**
O. OFFICE BUILDINGS	O1. Fireproof Up to Nine Stories	0.89
	O6. Bank Building	0.89
C. WALK UP APARTMENTS	C0. Three Families & Over	0.84
	C1. Over Six Families	0.78
	C2. Five to Six Families	0.85
R. CONDOMINIUMS	RC. Mixed Commercial/Condos	0.71
H. HOTELS	H1. Luxury Type, Built Prior to 1960 & Over	0.69
T. TRANSPORTATION FACILITIES	T2. Piers, Docks, Bulkheads	0.69
U. UTILITY BUREAU PROPERTIES	U6. Railroads, Private Ownership	0.67

TABLE 3
Predictability of different land uses

6.2 Periodicity of Taxi and Uber Demand

In order to reduce the number of plausible parameters in our search for optimal parameters for Markov and ARIMA predictors, we apply a standard Fourier transformation [36] to the taxi and the Uber demand time series, which also helps us understand the periodicity of the taxi and Uber demand:

$$M_k^{(i)} = \sum_{n=0}^{N-1} D_n^{(i)} e^{-i2\pi kn/N}. \quad (12)$$

Here $D_n^{(i)}$ is the n^{th} element of the i^{th} time series (data collected at the i^{th} location), $M_k^{(i)}$ is a k^{th} magnitude of the transformation (corresponding to k^{th} frequency), we employ it as a proxy measure for periodicity of a time series, N is the length of the time series and n iterates over the periods of all possible lengths for the given series.

Appendix Fig. S5 demonstrates the results of the transformation. Both of the data sets display similar periods: the highest magnitudes of the transformation correspond to the periods of 24, 12, 8 and 4 hours, which indicates that events separated by those time periods have the highest congruence. Using this information we select parameters for the Markov and the ARIMA algorithms that would take into account sub-sequences that cover the time period of 24, 12, 8 and 4 hours respectively.

6.3 Evaluation

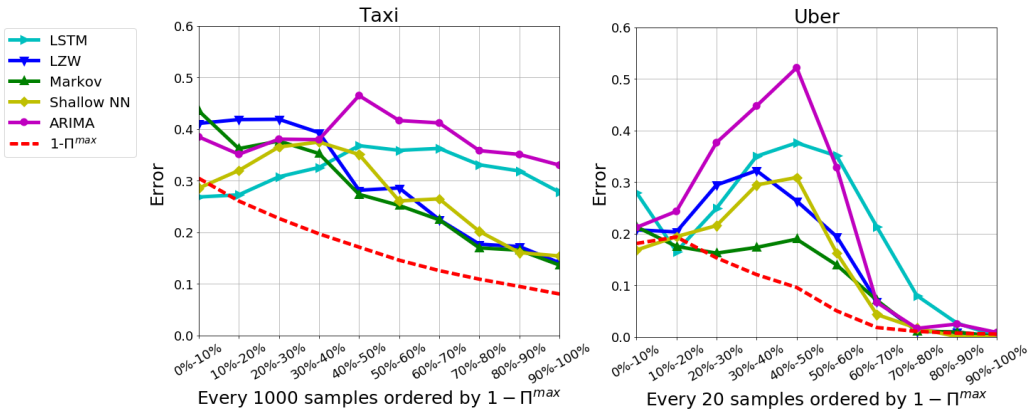
Prediction Error. We show the sMAPE errors of the five predictors and the prediction error bound $1 - \Pi^{max}$ in Fig. 6 (a) for the Yellow Taxi and Uber demand data respectively (see also Table 4 and 5 for the detailed results). In Table 4 and 5 we mark the prediction results with the lowest errors with the dark black color. We have ten building block groups with Π^{max} increasing from low to high (E.g., 0%-10% means the bottom 10% building block group according to Π^{max}).

We observe that, among all these predictors, when the maximum predictability Π^{max} is low, both the NN and the LSTM predictors provide better prediction accuracy (low sMAPE errors) for predicting the taxi demand (see Table 4). In the group with lowest maximum predictability $\Pi^{max} = 0.70$, the LSTM predictor has a prediction error 26.80%-the lowest of all predictors. The NN predictor has the second lowest prediction error 28.50%. We find similar results when predicting Uber demand (see Table 5). When there is a low predictability, both NN and LSTM predictors outperform other predictors.

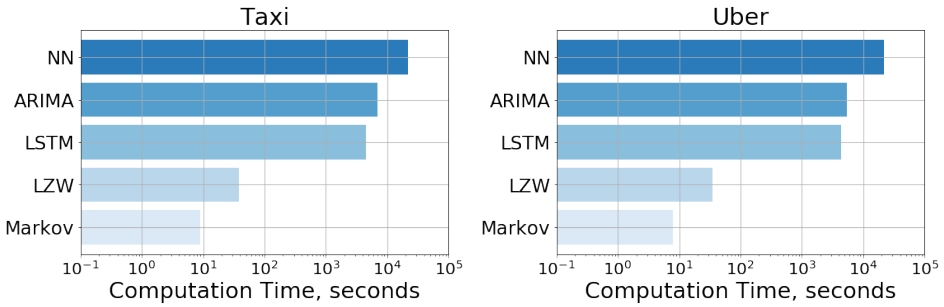
LSTM has been known for capturing the long-term dependency in the sequence modeling [20]. The NN predictor is able to capture the multiple features such as the weather information [10], which can not be captured by other predictors. That's why when there is a low predictability, by incorporating additional information such as weather information or mining hidden long-term temporal dependent patterns, we can achieve higher prediction accuracy using LSTM or NN predictor.

The Markov predictor provides better accuracy for building blocks with high predictability. In the building blocks with highest maximum predictability $\Pi^{max} = 0.92$, the Markov predictor is able to predict the taxi demand with an 13.60% prediction error (which equals to 86.40% prediction accuracy) for yellow taxi data, 1.8% better than the NN predictor and 14.1% better than the LSTM predictor. Besides, the computation time (see Fig. 6 (b)) of the Markov predictor is 2 seconds, only 0.02% of the NN predictor (about 2.7 hours). The Markov predictor is able to provide better predication accuracy with much less computation time for the areas with high predictability. We also observe that the Markov predictor converges to the predictability upper bound Π^{max} quickly. This is consistent with previous research [17] stating that there is a high positive correlation between the maximum predictability and the Markov predictor prediction accuracy.

Unlike the Markov predictor, the LZW predictor does not show a clear pattern when the maximum predictability is increasing. The reason is mainly because that the LZW predictor is based on the Lempel-Ziv-Welch text encoding algorithm and a LZW tree is maintained for the LZW predictor (see Appendix Alg. 2). The tree grows dynamically parsing the taxi demand history $D_n^{(i)}$. Thus the performance of the LZW predictor heavily depends how this subsequences LZW tree is built. That is the reason why the LZW tree performs so differently for the taxi and the Uber data sets. As it maintains a dynamic subsequences tree for the taxi demand prediction, if the tree is similar to the Markov predictor's lookup table, then its performance will be similar to the latter one. However, if the dynamic subsequences tree is unfortunately not optimally built, then its prediction accuracy will be much worse compared to the



(a)



(b)

Fig. 6. (a) Prediction errors, (b) Computation time of the Markov, LZW, NN, ARIMA and LSTM predictors.

Predictability	Π^{max} **(1-**Π^{max}**)**	**Markov**	**LZW**	**NN**	**ARIMA**	**LSTM**
0-10%	0.70 (30.40%)	43.50%	41.10%	28.50%	38.50%	**26.80%**
10-20%	0.74 (26.00%)	36.20%	41.80%	31.90%	35.10%	**27.20%**
20-30%	0.77 (22.70%)	37.70%	41.90%	36.50%	38.00%	**30.70%**
30-40%	0.80 (19.70%)	35.20%	39.30%	37.50%	37.90%	**32.50%**
40-50%	0.83 (17.10%)	**27.30%**	28.10%	35.00%	46.50%	36.80%
50-60%	0.85 (14.60%)	**25.20%**	28.60%	26.00%	41.60%	35.90%
60-70%	0.87 (12.50%)	**22.40%**	22.40%	26.40%	41.20%	36.20%
70-80%	0.89 (10.90%)	**16.90%**	17.70%	20.10%	35.80%	33.10%
80-90%	0.91 (9.46%)	**16.50%**	17.20%	16.00%	35.10%	31.90%
90-100%	0.92 (8.00%)	**13.60%**	14.10%	15.40%	33.00%	27.70%

TABLE 4
Prediction errors of the Markov, LZW, NN, ARIMA, and LSTM predictors on Yellow Taxi dataset at building block level

Predictability	Π^{max} **(1-**Π^{max}**)**	**Markov**	**LZW**	**NN**	**ARIMA**	**LSTM**
0-10%	0.81 (19.33%)	21.31%	20.71%	**16.78%**	21.18%	28.00%
10-20%	0.82 (18.07%)	17.52%	20.31%	19.40%	24.31%	**16.52%**
20-30%	0.85 (15.29%)	**16.20%**	29.41%	21.54%	37.63%	24.90%
30-40%	0.88 (12.04%)	**17.31%**	32.20%	29.42%	44.74%	34.99%
40-50%	0.92 (9.55%)	**18.91%**	26.32%	30.87%	52.15%	37.59%
50-60%	0.95 (5.00%)	**13.89%**	19.41%	16.20%	32.79%	35.05%
60-70%	0.98 (1.78%)	7.08%	6.89%	**4.30%**	6.65%	21.32%
70-80%	0.99 (1.02%)	**1.10%**	1.10%	1.65%	1.61%	7.96%
80-90%	0.993 (0.69%)	**0.88%**	0.88%	0.00%	2.45%	2.61%
90-100%	0.995 (0.49%)	**0.00%**	0.00%	0.00%	0.81%	0.13%

TABLE 5
Prediction errors of the Markov, LZW, NN, ARIMA, and LSTM predictors on Uber dataset

Markov one.

We apply a standard Fourier transformation [36] to the taxi and the Uber demand time series to find the best K for the order-K Markov predictor. Unlike the LZW predictor, the K in the Markov predictor is always optimized to capture the temporal pattern of the time series. Thus it is converging to the high prediction accuracy rapidly when the maximum predictability is increasing [17].

From the experiment we find that the maximum predictability Π^{max} can help us determining which predictor to use. In the areas with low predictability ($\Pi^{max} < 0.83$) we can use the NN and LSTM predictors to reach high accuracy by capturing the multiple features or hidden patterns. While in the areas with high predictability ($\Pi^{max} \geqslant 0.83$) the Markov predictor is able to provide better prediction accuracy while keeping the computation time low, as it is able converge to the predictability upper bound Π^{max} quickly and effecitently [17].

7 Related works

7.1 Predicting Taxi Demand

The taxi demand prediction problem has attracted more attention recently with the growth of the ride-sharing companies. Mukai et. al forecast the taxi demand from taxi historical data with a neural network (MLP) [10]. Li Et. al adapt the feature selection tool, L1-Norm SVM, to select the most salient feature patterns that determined the taxi performance [2]. An improved ARIMA based prediction method to forecast the spatio-temporal variation of passengers in hotspots is proposed by Li et. al [8]. Moreira-Matias et. al argued that ARIMA based prediction is not the best solution [7]. They proposed a new ensemble framework showing that their method can reach a high prediction accuracy. Li et al. [37] proposed a multi-task representation learning for taxi travel time estimation using the paths of historical trips during the training phase improving the performance.

The latest works use the deep neural networks combined with multiple features to predict the taxi demand in cities. Wang et. al [38] propose TaxiRec, a framework for evaluating and discovering the potential passengers of road clusters. TaxiRec includes three influential factors points-of-interest, road length and road type in their neural network predictive algorithm. Wang et. al [39] use a deep neural network structure to discover complicated taxi supply-demand patterns. They utilize multiple data sources including car-hailing orders, weather and traffic data. Different to these two papers, in this paper predict the taxi demand of each building block and we consider the taxi demand of one location as a time-series data. For one single building block, the road type, road length and point-of-interest types are fixed parameters and thus can not be considered as features for prediction here. By showing the predictability of different land use (Table 3), we show that the predictability is correlated with the point of interests, and thus corresponds to the prediction accuracy of different predictors.

7.2 Inferring Unmet Demand

From the taxi data set we can measure and predict the met taxi demand, that is, the number of the taxi services that emerged and will emerge at different locations. However, the unmet taxi demand, e.g., the number of people who need a taxi but could not find one, cannot be simply extracted from the taxi data set. To solve this problem, recent papers have tried to infer the unmet taxi demand from the taxi data set. In [12] the authors combined flight arrival with taxi demand and predict the passenger demand at different airport terminals in Singapore use queueing theory. Anwar et. al [11] formalized the unmet taxi demand problem and presented a novel heuristic algorithm to estimate it without any additional information. They inferred the unmet taxi demand from taxis with empty services and showed that it can be used to quantify the unmet demand. It must be noted that, although in our paper we only focus on predicting the met taxi demand , our method is a general solution and can be used for predicting unmet taxi demand.

7.3 Temporal Pattern of Human Mobility

It has been found that urban human mobility exhibits strong regularities, e.g., people usually go to work during daytime on weekdays, and go home after work. Marta et al. [40] found that the trajectories in human mobility exhibit strong regularities by studying cell phone user's locations. They show that human trajectories follow a high degree of temporal and spatial temporal pattern. Each person has a significant probability of returning to a few highly frequented locations such as home or working places. Wang et al. [41] model the time varying regularities of road traffic flows in road segments and intersections by mining statistic trajectories of all vehicles in the network, and design a routing algorithm for vehicle-to-vehicle data transmission in vehicular networks. Song et al. proposed the entropy-based probability to measure the temporal pattern of the individual human mobility [15]. They found a potential prediction algorithm that can reach up to 93% accuracy. They also observed that the user can be found in his or her most visited location during a corresponding hour long period with a high probability, which is indicative of a high temporal pattern of human mobility. The temporal pattern of human mobility also leads to the temporal pattern of taxi pick-ups. The human mobility patterns for different functional regions are different. Previous papers have analyzed the temporal pattern of urban human mobility and inferred the functions of the regions in three cities [42], [43]. Similar results can be found in Table 3, where there is a high predictability in the residential places and a low predictability in the transportation hubs.

8 Discussion

8.1 Effect of Spatial Resolution

We obtain the hourly number of taxi pick-ups per neighborhood, its maximum predictability, and the prediction errors of the five predictors and we examine the effect of the spatial resolution of the five predictors in Supplementary Figure S7, S8 and Supplementary Table SI. We have similar findings as the one we show in the evaluation section: when the predictability is high the NN predictors are able to achieve low prediction errors by capturing the hidden patterns, When the predictability is high we can use the simple

Markov predictor to achieve better prediction accuracy with much lower cost. The NN predictor is the best predictor when the maximum predictability of the hourly taxi pick-ups at the neighborhood level is low (among the bottom 30%). When the maximum predictability is high the Markov predictor easily outperforms the other four predictors (see Supplementary Table SI). The spatial resolution is not an important factor affecting the prediction performance of different predictors, the maximum predictability is.

8.2 Generality of Results

To test the generality of our findings, we run the experiments on the third spatial-temporal data set: the New York CitiBike data set (a bicycle-sharing usage data set) [4]. The data includes start station id, end station id, start time and end time for each bike trip. Overall, there are 328 stations which served as originating points for at least one bike ride with a grand total of 936,880 trips. We obtain the maximum predictability of the hourly bike usages per station and examine the performance of the five predictors. We show the results in Supplementary Table SII, Supplementary Figure S9 and S10.

For the stations where the bike usage has lower predictability, both LSTM and NN predictors are able to achieve lower prediction errors, as they are able to discover more complex non-linear patterns. On the other hand, for stations with higher predictability the Markov predictor is able to achieve low prediction errors with much less computation time. Therefore, having tested the maximum predictability and the predictors on three data sets that representing taxi demand, Uber demand and shared bike usage, we show that our finding is general: the maximum predictability can help determine which predictor to use in order to achieve low prediction errors and computational costs. It must be noted that the frequency of running a prediction model is dependent on the applications. For example, the prediction of health care application is usually in seconds [44], while other applications such as economic growth is usually in years [45]. In our paper we do the prediction of taxi demand at hourly and daily interval.

9 Conclusion

In this paper, we analyze over 14 million yellow and Uber taxi pick-up samples in NYC. We find that there is a high predictability of taxi demand (up to 83% in average), which indicates strong temporal correlation of human mobility. We also examine which commonly used predictive algorithm could approach the maximum predictability. We show that the compute-intensive deep learning predictor does not always have better prediction accuracy than the Markov one. In the areas with low predictability ($\Pi^{max} < 0.83$), the LSTM predictor can reach high accuracy by capturing the hidden long-term dependent temporal patterns. On the other hand, in the areas with high predictability ($\Pi^{max} \geqslant 0.83$), the Markov predictor is able to reach high prediction accuracy while keeping the computation time low. The temporal correlation of the taxi demand can be better captured in the Uber taxi data, possibly due to different cruising strategies.

It must be noted that our findings in this paper are not limited to the taxi or Uber demand problem. The same approach can be used for predictions in other time-series data sets [46], [47], such as the bike usage of a bike station [48] as shown above. We demonstrate that the knowledge of the predictability can help determine which predictor to use in the trade-off between the accuracy of a forecast and computational costs. Our finding is general and can be applied to other time-series data sets.

4. https://www.citibikenyc.com/system-data

Acknowledgment

The authors thank: the New York City TLC for providing the data used in this paper; Prof. Claudio Silva and Prof. Juliana Freire for their contribution on an early draft of this paper; and the anonymous reviewers for their insightful comments and suggestions. This work was supported in part by Alfred P. Sloan Foundation G-2018-11069, National Science Foundation Award 1827505, and Vingroup Innovation Award VINIF.2019.20.

References

[1] Y. Huang and J. W. Powell, "Detecting regions of disequilibrium in taxi services under uncertainty," in *SIGSPATIAL'12, Redondo Beach, CA, USA, November 7-9, 2012*, 2012, pp. 139–148.

[2] B. Li, D. Zhang, L. Sun, C. Chen, S. Li, G. Qi, and Q. Yang, "Hunting or waiting? discovering passenger-finding strategies from a large-scale real-world taxi dataset," in *IEEE PerCom , Seattle, WA, USA, Workshop Proceedings*, 2011, pp. 63–68.

[3] J. W. Powell, Y. Huang, F. Bastani, and M. Ji, "Towards reducing taxicab cruising time using spatio-temporal profitability maps," in *Proceedings of the 12th International Conference on Advances in Spatial and Temporal Databases*, ser. SSTD'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 242–260.

[4] R. Zhang and M. Pavone, "Control of robotic mobility-on-demand systems: A queueing-theoretical perspective," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 186–203, 2016.

[5] F. Miao, S. Han, S. Lin, J. A. Stankovic, D. Zhang, S. Munir, H. Huang, T. He, and G. J. Pappas, "Taxi dispatch with real-time sensing data in metropolitan areas: A receding horizon control approach," *IEEE Trans. Automation Science and Engineering*, vol. 13, no. 2, pp. 463–478, 2016.

[6] F. Miao, S. Han, S. Lin, Q. Wang, J. A. Stankovic, A. Hendawi, D. Zhang, T. He, and G. J. Pappas, "Data-driven robust taxi dispatch under demand uncertainties," *CoRR*, vol. abs/1603.06263, 2016.

[7] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi-passenger demand using streaming data," *IEEE Trans. Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1393–1402, 2013.

[8] X. Li, G. Pan, Z. Wu, G. Qi, S. Li, D. Zhang, W. Zhang, and Z. Wang, "Prediction of urban human mobility using large-scale taxi traces and its applications," *Frontiers of Computer Science in China*, vol. 6, no. 1, pp. 111–121, 2012.

[9] J. Yuan, Y. Zheng, L. Zhang, X. Xie, and G. Sun, "Where to find my next passenger," in *UbiComp 2011, Beijing, China*, 2011.

[10] N. Mukai and N. Yoden, *Taxi Demand Forecasting Based on Taxi Probe Data by Neural Network*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 589–597.

[11] A. Afian, A. Odoni, and D. Rus, "Inferring unmet demand from taxi probe data," in *ITSC 2013*, Sept 2015, pp. 861–868.

[12] A. Anwar, M. Volkov, and D. Rus, "Changinow: A mobile application for efficient taxi allocation at airports," in *ITSC 2013*, Oct 2013, pp. 694–701.

[13] New York City Taxi & Limousine Commission, "Taxi medallion increase – draft environmental impact statement," 2012. [Online]. Available: http://www.nyc.gov/html/tlc/downloads/pdf/taxi_deis_september_2013.pdf

[14] J. Xu, R. Rahmatizadeh, L. Blni, and D. Turgut, "Real-time prediction of taxi demand using recurrent neural networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2572–2581, Aug 2018.

[15] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.

[16] C. Zhong, E. Manley, S. M. Arisona, M. Batty, and G. Schmitt, "Measuring variability of mobility patterns from multiday smartcard data," *J. Comput. Science*, vol. 9, pp. 125–130, 2015.

[17] X. Lu, E. Wetter, N. Bharti, A. J. Tatem, and L. Bengtsson, "Approaching the Limit of Predictability in Human Mobility," *Scientific Reports*, vol. 3, Oct. 2013.

[18] C. T. Cheng, R. Jain, and E. van den Berg, "Mobile wireless systems: Location prediction algorithms," in *Encyclopedia of Wireless and Mobile Communications*, 2008.

[19] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley-Interscience, 2006.

[22] I. Kontoyiannis, P. H. Algoet, Y. M. Suhov, and A. J. Wyner, "Nonparametric entropy estimation for stationary processes and random fields, with applications to English text," *Information Theory, IEEE Transactions on*, vol. 44, no. 3, pp. 1319–1327, 1998.

[23] J. H. Wilkinson and J. H. Wilkinson, *The algebraic eigenvalue problem*. Clarendon Press Oxford, 1965, vol. 87.

[24] "Matlab Documentation. Root of nonlinear function." https://www.mathworks.com/help/optim/ug/fzero.html, accessed: 2019-07-20.

[25] R. P. Brent, "An algorithm with guaranteed convergence for finding a zero of a function," *The Computer Journal*, vol. 14, no. 4, pp. 422–425, 1971.

[26] L. Song, D. Kotz, R. Jain, and X. He, "Evaluating location predictors with extensive wi-fi mobility data," in *Proceedings IEEE INFOCOM 2004, Hong Kong, China, March 7-11, 2004*, 2004.

[27] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. The MIT Press, 2010.

[28] H. Bozdogan, "Model selection and akaike's information criterion (aic): The general theory and its analytical extensions," *Psychometrika*, vol. 52, no. 3, pp. 345–370, 1987.

[29] "New York polygon data set," https://www1.nyc.gov/site/planning/data-maps/open-data.page.

[30] A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *SIGMOD'84*, 1984, pp. 47–57.

[31] X. Xie, B. Mei, J. Chen, X. Du, and C. S. Jensen, "Elite: an elastic infrastructure for big spatiotemporal trajectories," *VLDB J.*, vol. 25, no. 4, pp. 473–493, 2016.

[32] F. J. Massey Jr, "The kolmogorov-smirnov test for goodness of fit," *Journal of the American statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.

[33] S. Makridakis and M. Hibon, "The m3-competition: results, conclusions and implications," *International Journal of Forecasting*, vol. 16, no. 4, pp. 451–476, 00 2000.

[34] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[35] K. Zhao, D. Khryashchev, J. Freire, C. Silva, and H. Vo, "Predicting taxi demand at high spatial resolution: Approaching the limit of predictability," in *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016, pp. 833–842.

[36] E. O. Brigham, E. O. Brigham, J. Rey Pastor, R. Pastor, T. M. T. M. Apostol, M. Rodríguez, M. R. Rodríguez, M. R. Martínez, C. H. Edwards, D. E. H. Edwards *et al.*, *The fast Fourier transform and its applications*. Prentice Hall,, 1988, no. 517.443.

[37] Y. Li, K. Fu, Z. Wang, C. Shahabi, J. Ye, and Y. Liu, "Multi-task representation learning for travel time estimation," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, 2018, pp. 1695–1704.

[38] R. Wang, C. Chow, Y. Lyu, V. C. S. Lee, S. Kwong, Y. Li, and J. Zeng, "Taxirec: recommending road clusters to taxi drivers using ranking-based extreme learning machines," in *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems, Bellevue, WA, USA, November 3-6, 2015*, 2015, pp. 53:1–53:4.

[39] D. Wang, W. Cao, J. Li, and J. Ye, "Deepsd: Supply-demand prediction for online car-hailing services using deep neural networks," in *33rd IEEE International Conference on Data Engineering, ICDE 2017, San Diego, CA, USA, April 19-22, 2017*, 2017, pp. 243–254.

[40] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, June 2008.

[41] Y. Wang, L. Huang, T. Gu, H. Wei, K. Xing, and J. Zhang, "Data-driven traffic flow analysis for vehicular communications," in *2014 IEEE Conference on Computer Communications, INFOCOM 2014, Toronto, Canada, April 27 - May 2, 2014*, 2014, pp. 1977–1985.

[42] K. Zhao, M. P. Chinnasamy, and S. Tarkoma, "Automatic city region analysis for urban routing," in *IEEE ICDMW 2015, Atlantic City, NJ, USA, November 14-17, 2015*, 2015, pp. 1136–1142.

[43] J. Yuan, Y. Zheng, and X. Xie, "Discovering regions of different functions in a city using human mobility and pois," in *KDD'12*, 2012, pp. 186–194.

[44] K. Srinivas, B. K. Rani, and A. Govrdhan, "Applications of data mining techniques in healthcare and prediction of heart attacks," *International Journal on Computer Science and Engineering (IJCSE)*, vol. 2, no. 02, pp. 250–255, 2010.

[45] A. Greiner, W. Semmler, and G. Gong, *The forces of economic growth: a time series perspective*. Princeton University Press, 2016.

[46] H. Wang, F. Xu, Y. Li, P. Zhang, and D. Jin, "Understanding mobile traffic patterns of large scale cellular towers in urban environment," in *Proceedings of the 2015 ACM Internet Measurement Conference, IMC 2015, Tokyo, Japan, October 28-30, 2015*, 2015, pp. 225–238.

[47] T. Cheng and T. Wicks, "Event detection using twitter: A spatio-temporal approach," *PLoS ONE*, vol. 9, pp. 1–10, 06 2014.

[48] D. Singhvi, S. Singhvi, P. Frazier, S. Henderson, E. Mahony, D. Shmoys, and D. Woodard, "Predicting bike usage for new york citys bike sharing system," 2015.

Kai Zhao Kai Zhao is an assistant professor at J. Mack Robinson College of Business, Georgia State University. His research interests are business analytics, data mining, mobile computing and text mining. Before coming to GSU, Dr. Kai Zhao worked as a post-doc associate at the New York University 2016-2017. He received his PhD in Computer Science from University of Helsinki, Finland in 2015, and his B.Sc in Computer Science from Shandong University, China in 2009 respectively. Dr. Kai Zhao has authored or co-authored 24 publications in refereed journals and prestigious conference presentations

Denis Khryashchev Denis Khryashchev is a Ph.D. student at the Graduate Center of the City University of New York. Previously he got his master's degree at the Center of Urban Science and Progress at the New York University where his main focus was on applying machine learning techniques on urban data sets. His main research interests are developing new algorithms and approaches for time series forecasting and analysis, and big data processing.

Huy Vo Huy T. Vo is an Assistant Professor of Computer Science at the City College of New York and a member of the doctoral faculty at the Graduate Center, City University of New York. He is also a faculty member at the Center for Urban Science and Progress, New York University. His current research focuses on high performance systems for interactive visualization and analysis of big data sets, specifically in urban applications. He has co-authored over 50 technical papers and 3 patents, and contributed to several widely-used open-source systems.

APPENDIX A
ALGORITHMS

Algorithm 1: Algorithm for Calculating Π^{max}

input : The values of $N^{(i)}$ and S, accuracy
output: The maximum predictability Π^{max}

// Support function to calculate i-th approximation.
$getApproximation(\Pi^{max}, N^{(i)}, S)$:
  // Value of the function (6).
  $f = f(\Pi^{max}, N^{(i)}, S)$;
  // Value of the first derivative.
  $d1 = \log_2\left(1-\Pi^{max}\right) - \log_2\left(\Pi^{max}\right) - \log_2\left(N^{(i)}-1\right)$;
  // Value of the second derivative.
  $d2 = \frac{1}{\left(\Pi^{max}-1\right)\Pi^{max}}$;
  return $\frac{f}{\left(d1 - f \times \frac{d2}{2d1}\right)}$;
// Check if the equation has no solutions (Lemma 2).
if $S > \log_2 N^{(i)}$;
then
  return "No solutions";
// Check if the solution approaches 1 (Lemma 5).
if $S \leq 0.01$;
then
  return 0.999;
// Else search between the maximum (Lemma 4) and 1.
$\Pi^{max} = (N^{(i)}+1)/(2N^{(i)})$;
// Iterate until the accuracy is achieved.
while $\left|f\left(\Pi^{max}, N^{(i)}, S\right)\right| > accuracy$ **do**
  $\Pi^{max} = \Pi^{max} - getApproximation\left(\Pi^{max}, N^{(i)}, S\right)$;
return Π^{max};

Algorithm 2: LZW Predictor

input : The taxi demand time series $D_n^{(i)}$
output: The predicted value of taxi demand β at the time $n+1$

Set buffer subsequence equal to the first value in $D_n^{(i)}$, $b = d_1^{(i)}$;
Initialize an empty LZW Tree;
Initialize $j = 2$;
while $j \leq n$;
do
  if *b followed by* $d_j^{(i)}$ *is in the LZW tree;*
  then
    Append $d_j^{(i)}$ to the buffer b;
  else
    Update the LZW tree with b followed by $d_j^{(i)}$;
    Update $count\left(d_{t\leq j}^{(i)}\right)$ at each node;
    Set $b = d_j^{(i)}$;
  Set $j = j+1$;
Find the subsequence $s_m^{(i)}$ followed by β with the highest probability in the the LZW tree;
return β;

Algorithm 3: NN Predictor

input : The taxi demand time series $D_n^{(i)}$ combined with exogenous variables, corresponding time stamps $T = \{t_1, \ldots t_n\}$
output: The predicted value of future taxi demand $\beta^{'}$ at the time $n+1$

Create additional binary features from the time stamps: each of the time periods and dates is assigned an individual binary feature that turns on during the period and date.;
Train a separate Neural Network for each building block and predict the future value of taxi demand β;
Calculate smoothed prediction $\beta^{'}$ with linear combination of β and m latest values of taxi demand with Gaussian kernel. return $\beta^{'}$;

Algorithm 4: ARIMA Predictor

input : The taxi demand time series $D_n^{(i)}$ from time stamps $T = \{1, \ldots, n\}$ per building block
output: The predicted taxi demand β at time stamps $n+1$

Calculating AIC scores for each tuple (p_i, d_i, q_i) from the parameter set (P, D, Q);
Selecting a tuple (p_k, d_k, q_k) that has the smallest AIC score;
Fitting the ARIMA(p_k, d_k, q_k) model for $D_n^{(i)}$;
return $\hat{D}_{n+1}^{(i)}$ as β;

Algorithm 5: Order $O(k)$ Markov chain Predictor

input : The taxi demand time series $D_n^{(i)}$, order of the predictor k, and the Markov chain Predictor matrix T
output: The predicted taxi demand β at time $n+1$

Extracting the recent subsequence of length k, $c = d_{n-k+1}^{(i)} d_{n-k+2}^{(i)} \ldots d_n^{(i)}$ from $D_n^{(i)}$ ending at time n;
Looking for the subsequence c in the matrix T at row $T(c)$;
if $T(c)$ *is empty* // The taxi demand time series c did not occur previously.
then
  return the taxi demand with the highest probability in $D_n^{(i)}$;
else
  Finding the subsequence c followed by β with the highest probability at the row $T(c)$;
  // Similar taxi demand history has appeared at least once before
  return β;
Update the matrix T with the new taxi demand $d_{n+1}^{(i)}$ at the time $n+1$.

A.1 ARIMA model

Formaly ARIMA model is defined as

$$\left(1 - \sum_{i=1}^{p} \phi_i L^i\right)(1-L)^d d_t^{(i)} = \delta + \left(1 + \sum_{i=1}^{q} \theta_i L^i\right)\epsilon_t \quad (13)$$

where L^k is a lag operator L of order k, $L^k d_t^{(i)} = d_{t-k}^{(i)}$, $d \in \mathbb{N}$ represents the multiplicity of a unit root $(1-L)$ usually denoted as integration, δ introduces the drift of the model that is equal to $\delta / \sum_i \phi_i$, p and q specify the number of the parameters ϕ of the auto-regressive, and θ of the moving average components of the model correspondingly, ϵ_t is the normally distributed error.

Order of the model can be denoted as ARIMA(p, d, q), where p is the number of auto-regressive terms, d is the number of nonseasonal differences (integration) that introduce nonstationarity when $d \neq 0$, and q is the number of error terms (moving-average).

A standard way to identify the proper order of the model is to the Akaike Information Criterion (AIC) [1]:

$$AIC(p, d, q) = -2\log(\mathcal{L}) + 2(p+q+k+1) \quad (14)$$

where $\mathcal{L}$ is the maximum likelihood estimator of the innovation variance, $k \in \{0, 1\}$ stands for the presence or absence of the constant term in the model, and the sum $p+q+k+1$ represents the total number of parameters in the model.

The optimal model ARIMA(p, d, q) is selected as the one that has the lowest AIC coefficient compared with the other orders considered. To improve the prediction accuracy and decrease the number of overall iterations, one might consider relating the choice of the parameters p and q with the inverse frequencies of largest magnitudes in the frequency spectrum of the original time series X. Alternative approaches to select the parameters (p, d, q) minimize $\sum_{i=1}^{N} \epsilon_i^2$.

A.2 LSTM

The inputs for the model enter LSTM cells that contain 3 gates within their structure: input, output, and forget. In the input gate the original input $D_n^{(i)}$ is first passed through the layers with σ and $\tanh$ activation functions:

$$i_1 = \tanh\left(b^{i_1} + D_n^{(i)} W_1^{i_1} + h_{n-1} W_2^{i_1}\right) \tag{15}$$

$$i_2 = \sigma\left(b^{i_2} + D_n^{(i)} W_1^{i_2} + h_{n-1} W_2^{i_2}\right) \tag{16}$$

where b^{i_1} and b^{i_2} are the input biases, $W_1^{i_1}$ and $W_1^{i_2}$ stand for the input weights, $W_2^{i_1}$ and $W_2^{i_2}$ are the weights of the previous cell's output h_{n-1}. The output of the input gate is the result of pointwise multiplication of the outputs of the $\tanh$ and σ layers, namely $i_1 \otimes i_2$.

The forget gate is evaluated by another neural network layer with sigmoid activation

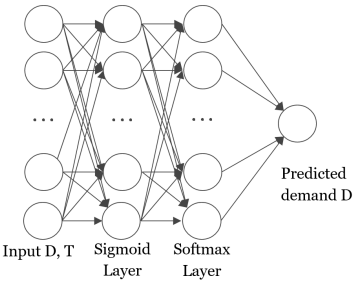
$$f = \sigma\left(b^f + D_n^{(i)} W_1^f + h_{n-1} W_2^f\right) \tag{17}$$

Therefore, the combined output of the input and forget gates is $f + i_1 \otimes i_2$.

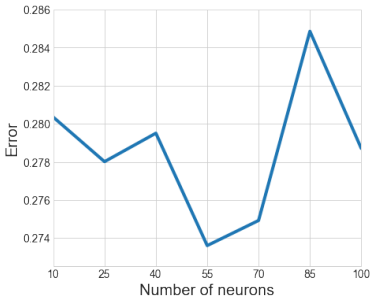
The output gate is yet another layer with σ activation

$$o = \sigma\left(b^o + D_n^{(i)} W_1^o + h_{n-1} W_2^o\right) \tag{18}$$

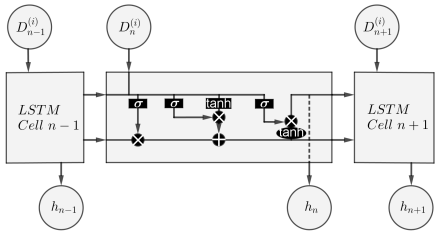
Finally, the output of the entire LSTM cell is calculated by $h_n = \tanh\left(f + i_1 \otimes i_2\right) \otimes o$.



Supplementary Figure S2. MLP neural network preditor

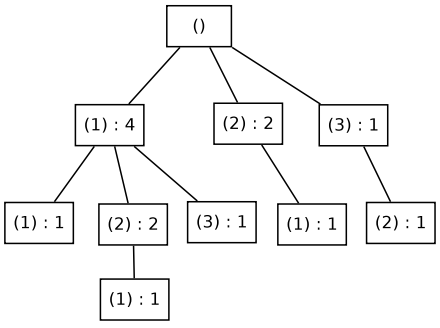


Supplementary Figure S3. sMAPE errors given the number of neurons per layer



Supplementary Figure S4. LSTM preditor

APPENDIX B
FIGURES AND TABLES



D = 11212132

s = 1, 2, 3, 11, 12, 21, 121, 13, 32

Supplementary Figure S1. LZW tree

Predictability	Π^{max} (1-Π^{max})	Markov	LZW	NN	ARIMA	LSTM
0-10%	0.92 (7.95%)	23.02%	14.73%	**13.56%**	17.81%	19.24%
10-20%	0.93 (6.85%)	11.26%	12.75%	13.15%	**10.77%**	11.23%
20-30%	0.94 (6.01%)	18.53%	21.10%	**14.76%**	18.19%	21.19%
30-40%	0.948 (5.16%)	**13.07%**	20.27%	19.97%	21.49%	23.84%
40-50%	0.96 (3.92%)	**15.64%**	37.20%	28.12%	46.74%	23.65%
50-60%	0.973 (2.66%)	**15.99%**	19.30%	18.58%	28.50%	18.27%
60-70%	0.987 (1.29%)	6.94%	**6.28**%	6.78%	6.90%	9.82%
70-80%	0.995 (0.46%)	**0.33%**	0.33%	0.33%	0.00%	4.17%
80-90%	0.996 (0.39%)	**0.00%**	0.00%	0.00%	0.00%	1.19%
90-100%	0.996 (0.38%)	**0.00%**	0.00%	0.17%	5.58%	4.47%

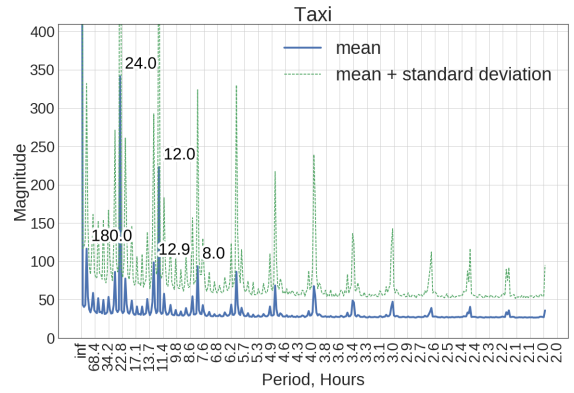
Supplementary Table SI
Prediction errors of the Markov, LZW, NN, ARIMA, and LSTM predictors on Yellow Taxi dataset at neighborhood level

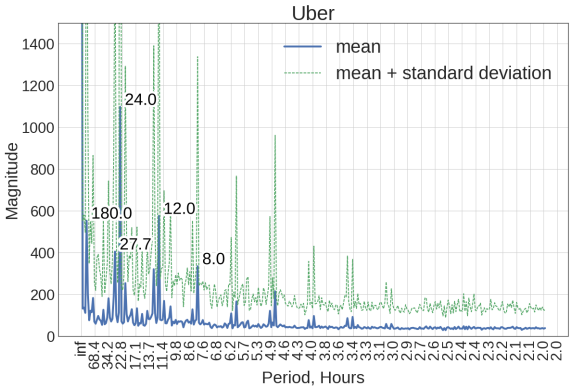
Predictability	Π^{max} (1-Π^{max})	Markov	LZW	NN	ARIMA	LSTM
0-10%	0.72 (27.54%)	53.87%	47.82%	**28.89%**	48.51%	31.23%
10-20%	0.74 (26.20%)	51.74%	46.32%	**30.32%**	43.77%	34.62%
20-30%	0.75 (24.88%)	51.30%	45.50%	**28.60%**	45.19%	31.54%
30-40%	0.76 (23.66%)	45.26%	48.50%	**30.76%**	41.49%	33.37%
40-50%	0.78 (21.94%)	44.39%	43.60%	**30.96%**	50.74%	36.85%
50-60%	0.79 (20.56%)	39.00%	42.24%	**32.57%**	45.15%	40.01%
60-70%	0.81 (19.26%)	37.00%	38.15%	**34.25%**	38.50%	37.83%
70-80%	0.83 (16.96%)	30.79%	35.14%	**29.16%**	37.39%	35.58%
80-90%	0.85 (14.98%)	**23.62%**	28.56%	23.24%	35.80%	30.43%
90-100%	0.91 (8.72%)	**21.59%**	26.12%	21.19%	33.24%	29.64%

Supplementary Table SII
Prediction errors of the Markov, LZW, NN, ARIMA, and LSTM predictors on Citibike dataset

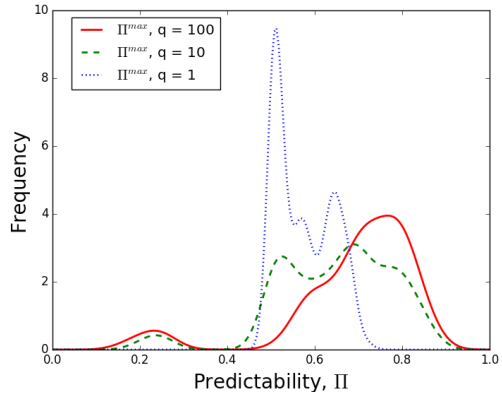
Predictability	no lag	lag 1	lag 2	lag 3	lag 4	lag 8	lag 12	lag 24
0.694	**3e-06**	0.669	0.179	0.231	**0.036**	**0.024**	0.065	**0.017**
0.739	**0.042**	0.952	0.537	0.928	0.597	0.308	0.584	0.657
0.774	**8e-13**	0.08	0.058	0.07	0.067	**0.04**	**0.023**	**0.008**
0.803	0.225	0.958	0.782	0.959	0.918	0.926	0.933	0.998
0.829	0.995	1.0	0.964	0.933	0.614	0.587	0.347	0.444
0.854	0.102	0.826	0.169	0.166	0.074	**0.038**	0.117	0.067
0.875	1.0	0.979	0.989	0.897	0.905	0.798	0.497	0.342
0.891	0.589	0.962	0.947	0.869	0.753	0.638	0.757	0.426
0.906	0.97	1.0	0.975	0.999	0.991	0.964	0.875	0.867
0.922	0.527	0.995	0.185	0.349	**0.001**	**0.004**	**0.002**	**0.028**

Supplementary Table SIII
Results of two sample KS test for the distributions of sample time series vs the rest with $\alpha = 0.05$. p values $< \alpha$ shown in bold.

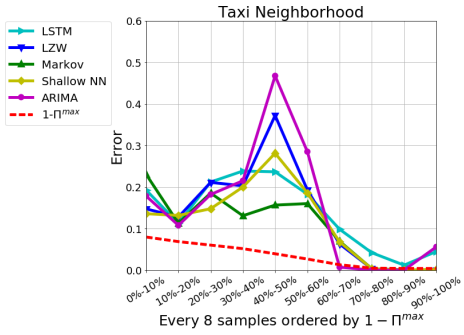




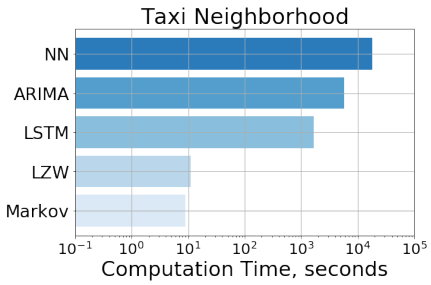
Supplementary Figure S5. Fourier transform with periods of the strongest magnitude for taxi and Uber data sets



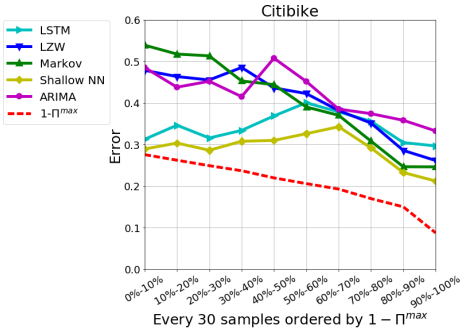
Supplementary Figure S6. Effect of of categorize size q



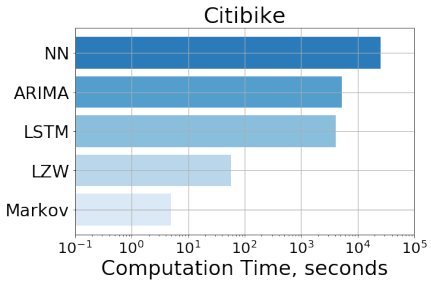
Supplementary Figure S7. Prediction errors of the Markov, LZW, NN, ARIMA and LSTM predictors at the neighborhood level.



Supplementary Figure S8. Neighborhood prediction time.



Supplementary Figure S9. Prediction errors of the Markov, LZW, NN, ARIMA and LSTM predictors using citibike data.



Supplementary Figure S10. Citi Bike prediction time

APPENDIX C
PROOF OF THE MAXIMUM PREDICTABILITY Π^{max}

The maximum predictability Π^{max} can be solved by the following equation:

$$S^{(i)} = -\Pi^{max} log_2(\Pi^{max}) - (1-\Pi^{max}) log_2(1-\Pi^{max}) + (1-\Pi^{max}) log_2(N^{(i)}-1)$$

Our proof, which is adopted from [2], is provided below.

Proof. Given any predictive algorithm α, let $P_\alpha(X_n^{(i)} = \hat{X}_n^{(i)}|h_{n-1}^{(i)})$ be the distribution generated over the next possible taxi demand $\hat{X}_n^{(i)}$ at the location i. Here $h_{n-1}^{(i)} = \{X_{n-1}^{(i)}, X_{n-2}^{(i)}, \ldots, X_1^{(i)}\}$ be the location i 's past taxi demand from time 1 to $n-1$. Let $X_t^{(i)}$ represents the real demand at the time t. $Pr[X_n^{(i)} = x^{(i)}|h_{n-1}^{(i)}]$ is the probability that the next taxi demand $X_n^{(i)}$ is $x^{(i)}$, given the taxi demand history $h_{n-1}^{(i)}$. Thus $P(X_n^{(i)}|h_{n-1}^{(i)})$ is the true distribution over the next taxi demand.

Let $\pi(h_{n-1}^{(i)})$ be the probability that there is a most likely taxi demand at the location i given the taxi demand history $h_{n-1}^{(i)}$. We have

$$\pi(h_{n-1}^{(i)}) = sup_x\{Pr[X_n^{(i)} = x|h_{n-1}^{(i)}]\},$$

The probability of successfully predicting the next taxi demand is $Pr_\alpha\{X_n^{(i)} = \hat{X}_n^{(i)}|h_{n-1}^{(i)}\} = \sum_{x^{(i)}} P(x^{(i)}|h_{n-1}^{(i)}) P_\alpha(x^{(i)}|h_{n-1}^{(i)})$. Since $\pi(h_{n-1}^{(i)}) \geq P(x^{(i)}|h_{n-1}^{(i)})$ for any $x^{(i)}$, we have

$$\begin{aligned} Pr_\alpha\{X_n^{(i)} = \hat{X}_n^{(i)}|h_{n-1}^{(i)}\} &= \sum_{x^{(i)}} P(x^{(i)}|h_{n-1}^{(i)}) P_\alpha(x^{(i)}|h_{n-1}^{(i)}) \\ &\leq \sum_{x^{(i)}} \pi(h_{n-1}^{(i)}) P_\alpha(x^{(i)}|h_{n-1}^{(i)}) \\ &= \pi(h_{n-1}^{(i)}) \sum_x^{(i)} P_\alpha(x^{(i)}|h_{n-1}^{(i)}) \\ &= \pi(h_{n-1}^{(i)}) \end{aligned}$$

Then we define the predictability $\Pi(n)$ for a taxi demand series with a history of length $n-1$. $P(h_{n-1}^{(i)})$ represents the probability of a particular taxi demand history $h_{n-1}^{(i)}$. If we sum over all the possible histories of length $n-1$, we have the predictability as $\Pi(n) = \sum_{h_{n-1}^{(i)}} P(h_{n-1}^{(i)}) \pi(h_{n-1}^{(i)})$. The overall predictability Π can be defined as $\Pi = \lim_{n\to\infty} \frac{1}{n} \sum_t^n \Pi(t)$.

Let $N^{(i)}$ be the total number of possible taxi demand and there is a uniform distribution over the remaining $N^{(i)}-1$ possible taxi demand. Then we will have X', whose distribution $P'(X^{(i)}|h^{(i)}) = (p, \frac{1-p}{N^{(i)}-1}, \frac{1-p}{N^{(i)}-1} \ldots, \frac{1-p}{N^{(i)}-1})$. Note here $S(X_n^{(i)}|h_{n-1}^{(i)}) \leq S(X'|H_{n-1}^{(i)})$. Then we have

$$\begin{aligned} S(X') &= -plog_2(p) - \sum \frac{1-p}{N^{(i)}-1} log_2(\frac{1-p}{N^{(i)}-1}) \\ &= -plog_2(p) - (1-p) log_2(\frac{1-p}{N^{(i)}-1}) \\ &= -[plog_2 p + (1-p) log_2(1-p)] \\ &\quad + (1-p) log_2(N^{(i)}-1) \\ &= S_F(p) \\ &= S_F(\pi(h_{n-1}^{(i)}) \end{aligned}$$

Note here the Fano function $S_F(p)$ is concave and monotonically decreases with p. Based on Fano's inequality, $S(X_n^{(i)}|h_{n-1}^{(i)}) \leq S_F(\pi(h_{n-1}^{(i)}))$. Following Jensen's inequality, we have

$$\begin{aligned} S(n) &= \sum_{h_{n-1}^{(i)}} P(h_{n-1}^{(i)}) S(X_n^{(i)}|h_{n-1}^{(i)}) \\ &\leq \sum_{h_{n-1}^{(i)}} P(h_{n-1}^{(i)}) S_F(\pi(h_{n-1}^{(i)})) \\ &\leq S_F\left(\sum_{h_{n-1}^{(i)}} P(h_{n-1}^{(i)}) \pi(h_{n-1}^{(i)})\right) \\ &= S_F(\Pi(n)) \end{aligned}$$

For a stationary stochastic process $\chi = X_t^{(i)}$, based on $S(n) \leq S_F(\Pi(n))$ and Jensen's inequality, we have

$$\begin{aligned} S &= \lim_{n\to\infty} \frac{1}{n} S(X_1^{(i)}, X_2^{(i)}, \ldots, X_n^{(i)}) \\ &= \lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^n S(X_t^{(i)}|h_{t-1}^{(i)}) \\ &= \lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^n S(t) \\ &\leq \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^n S_F(\Pi(t)) \\ &\leq S_F\left(\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^n (\Pi(t))\right) \\ &= S_F(\Pi) \end{aligned}$$

Here $S = \lim_{n\to\infty} \frac{1}{n} S(X_1^{(i)}, X_2^{(i)}, \ldots, X_n^{(i)})$ is the definition of entropy. We define $S(t) = S(X_t^{(i)}|h_{t-1}^{(i)})$ as the conditional entropy at the time t. Let Π^{max} be the solution to the equation $S = S_F(\Pi^{max}) \leq S_F(\Pi)$. Since $S_F(p)$ is concave and monotonically decreased, we have $\Pi \leq \Pi^{max}$, which means that Π^{max} is the upper bound of the predictability Π. The predictability upper bound Π^{max} can be solved by solving the following equation:

$$\begin{aligned} S &= S_F \Pi^{max} \\ &= -\Pi^{max} log_2(\Pi^{max}) - (1-\Pi^{max}) log_2(1-\Pi^{max}) \\ &\quad + (1-\Pi^{max}) log_2(N^{(i)}-1) \end{aligned}$$

□

APPENDIX D
PROPERTY OF THE MAXIMUM PREDICTABILITY Π^{max}

The maximum predictability Π^{max} can be obtained by solving Equation 5. In this equation, both S and $N^{(i)}$ are known numbers. The equation is transcendental, i.e. it does not have a closed form solution. But it can be approximated using numerical methods and analysis. We give five lemmas on the relations between $N^{(i)}$, S, and Π^{max} to help calculate the maximum predictability:

LEMMA 1. *$f(\Pi^{max}) = 0$ has one critical point–the maximum at $\Pi^{max} = \frac{1}{N^{(i)}}$: it monotonically decreases if $\Pi^{max} > \frac{1}{N^{(i)}}$ and increases if $\Pi^{max} < \frac{1}{N^{(i)}}$.*

PROOF. Given that $\log_2 \Pi^{max} = \frac{\ln \Pi^{max}}{\ln 2}$ and $S \ln 2$ is a constant, the derivative of the function is $\frac{\partial f(\Pi^{max})}{\partial \Pi^{max}} = \frac{\left(-\ln \Pi^{max} + \ln\left(1-\Pi^{max}\right) - \ln\left(N^{(i)}-1\right)\right)}{\ln 2}$. To find the critical points of the function we equal it to zero $\frac{\partial f(\Pi^{max})}{\partial \Pi^{max}} = 0$ and we get $\Pi^{max} = \frac{1}{N^{(i)}}$. Therefore, the function always has only one critical point at $\frac{1}{N^{(i)}}$.

To examine the behavior of Π^{max} near the point $\frac{1}{N^{(i)}}$, we examine $\Pi^{max} = \frac{1}{N^{(i)} \pm \varepsilon}$ where $\varepsilon > 0$ is an arbitrary infinitely small positive number. We obtain two expressions for the addition and subtraction of ε that correspond to the left and the right side of the maximum: $-\ln\left(\frac{1}{N^{(i)}} + \varepsilon\right) + \ln\left(1 - \left(\frac{1}{N^{(i)}} + \varepsilon\right)\right) - \ln\left(N^{(i)} - 1\right)$ and $-\ln\left(\frac{1}{N^{(i)}} - \varepsilon\right) + \ln\left(1 - \left(\frac{1}{N^{(i)}} - \varepsilon\right)\right) - \ln\left(N^{(i)} - 1\right)$.

After simplification, the first expression converts into $-(N^{(i)})^2\varepsilon$. Because $(N^{(i)})^2 > 0$ and $\varepsilon > 0$, we derive that $-(N^{(i)})^2\varepsilon < 0$, therefore, $\frac{\partial f\left(\Pi^{max} = \frac{1}{N^{(i)}+\varepsilon}\right)}{\partial \Pi^{max}} < 0$. The function monotonically decreases to the right of the critical point at $\frac{1}{N^{(i)}}$.

The second expression can be simplified as $(N^{(i)})^2\varepsilon$. Because $(N^{(i)})^2$ is larger than 0 and ε is larger than 0, we obtain $(N^{(i)})^2\varepsilon > 0$. Therefore, $\frac{\partial f\left(\Pi^{max} = \frac{1}{N^{(i)}-\varepsilon}\right)}{\partial \Pi^{max}} > 0$. The function monotonically increases to the left of the critical point at $\frac{1}{N^{(i)}}$, and the critical point is the maximum value.

LEMMA 2. *$f(\Pi^{max}) = 0$ has no solutions if $N^{(i)} < 2^S$.*

PROOF. We prove it by contradiction. Supposing the function has solutions for $N^{(i)} < 2^S$. From Lemma 1 we know that the function always has a global maximum at $\Pi^{max} = \frac{1}{N^{(i)}}$. The value of the function in the global maximum is $f(\Pi^{max} = \frac{1}{N^{(i)}}) = log_2(N^{(i)}) - S$. If $N^{(i)} < 2^S$, $N^{(i)} = 2^S - \varepsilon$, $\varepsilon > 0$ then $f(\Pi^{max} = 2^S - \varepsilon) = log_2(2^S - \varepsilon) - S < 0$. Because the maximum value is below 0 that implies that all the other values are below zero as well and the equation has no solutions, $\{\Pi^{max} | f(\Pi^{max}) = 0\} = \varnothing$.

LEMMA 3. *$f(\Pi^{max}) = 0$ has exactly one solution if $N^{(i)} = 2^S$ and the solution is $\frac{1}{N^{(i)}}$.*

PROOF. Following the results of Lemma 2, the value of the function at the maximum is $f(\Pi^{max} = \frac{1}{N^{(i)}}) = log_2(N^{(i)}) - S$. Setting it equal to 0 we obtain $N^{(i)} = 2^S$, i.e. the maximum lies on the axis.

LEMMA 4. *$f(\Pi^{max}) = 0$ has at most two solutions if $N^{(i)} > 2^S$ and the biggest solution is in $\left(\frac{1}{N^{(i)}}, 1\right)$.*

PROOF. Following the results of Lemma 3, we know that the function has only one global maximum $max(f(\Pi^{max})) = f(\Pi^{max} = \frac{1}{N^{(i)}}) = log_2(N^{(i)}) - S$. Having $N^{(i)} > 2^S$ we analyze the function at $N^{(i)} = 2^S + \varepsilon$, $\varepsilon > 0$. Since $f(\Pi^{max} = 2^S + \varepsilon) = log_2(2^S + \varepsilon) - S > 0$, the only maximum of the function is strictly above zero. Because the function is continuous, it has two solutions at most for $\Pi^{max} > 0$. The solutions are located to the left and to the right of the maximum, therefore, the bigger solution is bounded by $(\frac{1}{N^{(i)}}, 1)$.

LEMMA 5. *If $S \to 0$, the solution to $f(\Pi^{max}) = 0$ approaches 1.*

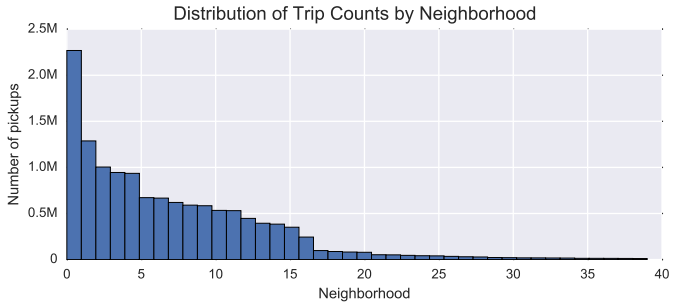
PROOF. At first, we examine what happens when $S = 0$. The equation turns into $-\Pi^{max} ln(\Pi^{max}) - (1 - \Pi^{max}) \times ln(1 - \Pi^{max}) + (1 - \Pi^{max}) ln(N^{(i)} - 1) = 0$. Grouping the similar terms together around $(1 - \Pi^{max})$ we obtain $(1 - \Pi^{max}) ln\left(\frac{\Pi^{max}(N^{(i)}-1)}{1-\Pi^{max}}\right) - ln(\Pi^{max}) = 0$. Expanding the brackets for $(1 - \Pi^{max})$, grouping the logarithms together and performing exponentiation we get $\left(\frac{N^{(i)}-1}{\frac{\Pi^{max}}{1-\Pi^{max}}(1-\Pi^{max})}\right)^{1-\Pi^{max}} = 1$. Having 1 on the right side means that $1 - \Pi^{max} = 0$ and therefore, $\Pi^{max} = 1$.

Second, we repeat the procedure with $S > 0$. We obtain $\left(\frac{\Pi^{max}\frac{2}{1-\Pi^{max}}(N^{(i)}-1)}{1-\Pi^{max}}\right)^{1-\Pi^{max}} = 2^{S \cdot ln(2)}$. Having the $\lim_{S\to 0}(2^{S \cdot ln(2)}) = 2^0 = 1$ implies that the entire function becomes 1 as S approaches 0.

To help the readers better understand the property of the maximum predictability (Lemma 1-5), in Fig. 2 we demonstrate different behavior of the equation given different pairs of parameters $N^{(i)}$ and S. Note here circles indicate maximums, and rectangles indicate solutions. For $N^{(i)} = 6$ and $S = 1.2$ we observe the solution at $\Pi^{max} = 0.8$, for $N^{(i)} = 2$ and $S = 0.1$ we obtain $\Pi^{max} = 0.99$, for $N^{(i)} = 4$ and $S = 2$ function turns into a zero at $\Pi^{max} = 0.25$, and finally, for $N^{(i)} = 6$ and $S = 3$ there are no solutions ($\{\Pi^{max} | f(\Pi^{max}) = 0\} = \varnothing$).

APPENDIX E
EFFECT OF CATEGORIZING FACTOR

Since there can be high variability in taxi or Uber demand, it is hard to predict the exact value for $d_t^{(i)}$. For simplicity, we use a factor q to round the value for the taxi demand. We group every q taxi demand as one taxi demand. We try to predict the taxi demand at every q level, e.g., if q = 10, then 620-629 all become 620 in the sequence. After a few tests, we set $q = 10$ for making the prediction more relaxed while keeping the errors low: in Supplementary Fig. S6 we study the effect of the category size q. The red dashed line, the green dashed line and the black dashed line refer to the probability density function of Π^{max} when q equals to 100, 10 and 1 respectively. When q equals to 1, it means that there is no categorization in terms of prediction. As shown in the figure, we can find that with an increasing q, the upper bound of predictability Π^{max} goes up. It means that an potential prediction algorithm has higher prediction accuracy at every 100 level compared to at every 10 or 1 level. We set $q = 10$ and make the prediction more relaxed while keeping the prediction errors low.



Supplementary Figure S11. Heterogeneous taxi demand at different regions ranging between a few and a few million pickups.

APPENDIX F
MAP PARTITION METHODS

In this paper we propose a method of predicting the taxi demand at high spatial resolution, i.e., predicting the taxi demand at building block level. In many previous works [3], [4], [5], [6], [7], the authors usually partition a city into grids or neighborhoods, e.g. partitioning NYC into equal-size grids [3].

There are two problems with the grid or neighborhood based partition method. First, grid or neighborhood based partition method can lead to inaccurate results. It has been found that the population of a city is exponentially distributed, decreasing from the city center to rural areas [8]. Grid or neighborhood based partition method will lead to under-fit for the prediction of the taxi demand service in the city center, while over-fit occurs for the prediction of the taxi demand service in rural areas.

In Fig S11. we partition NYC into 40 neighborhoods and plot the histogram of the taxi pick-ups of these neighborhoods. We observe that there are big differences in terms of the taxi pick-ups in different neighborhoods. Some neighborhoods contain a lot of pick-ups, while other neighborhoods contain few. The detailed information of taxi demand in the highly dense neighborhoods will be hidden.

The second problem is that a grid or neighborhood may hide the rich information of the city in dense urban areas. Even a relatively high resolution grid may contain mismatch structures, e.g. overlapping highways, across governmental units and boundaries with very different characteristics. Physical patterns of city development such as zoning and land-use applications is a richer data source, and is often said to be a good context feature for machine-learning predictors [7], [6], [9]. One building block with a single function might be mapped to multiple grids, which will lead to inaccurate results for machine-learning predictors.

To overcome the limitation of the grid-based or neighborhood-based partition methods, in this paper we propose to predict the taxi demand at the high spatial resolution, i.e., prediction at building block level. The prediction of taxi demand at the building block level did not only address the issues of the under-fit of predicting taxi demand service at the city center, but also provides additional information such as the land-use characteristics of the building block, which can be used for multiple feature machine learning algorithms [9], [4].

REFERENCES

[1] H. Bozdogan, "Model selection and akaike's information criterion (aic): The general theory and its analytical extensions," *Psychometrika*, vol. 52, no. 3, pp. 345–370, 1987.
[2] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.
[3] F. Miao, S. Han, S. Lin, Q. Wang, J. A. Stankovic, A. Hendawi, D. Zhang, T. He, and G. J. Pappas, "Data-driven robust taxi dispatch under demand uncertainties," *CoRR*, vol. abs/1603.06263, 2016.
[4] N. Mukai and N. Yoden, *Taxi Demand Forecasting Based on Taxi Probe Data by Neural Network*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 589–597.
[5] J. Yuan, Y. Zheng, L. Zhang, X. Xie, and G. Sun, "Where to find my next passenger," in *UbiComp 2011, Beijing, China*, 2011.
[6] J. Yuan, Y. Zheng, and X. Xie, "Discovering regions of different functions in a city using human mobility and pois," in *KDD'12*, 2012, pp. 186–194.
[7] K. Zhao, M. P. Chinnasamy, and S. Tarkoma, "Automatic city region analysis for urban routing," in *IEEE ICDMW 2015, Atlantic City, NJ, USA, November 14-17, 2015*, 2015, pp. 1136–1142.
[8] X. Liang, J. Zhao, L. Dong, and K. Xu, "Unraveling the origin of exponential law in intra-urban human mobility," *Sci. Rep.*, vol. 3, Oct 2013.
[9] B. Li, D. Zhang, L. Sun, C. Chen, S. Li, G. Qi, and Q. Yang, "Hunting or waiting? discovering passenger-finding strategies from a large-scale real-world taxi dataset," in *IEEE PerCom , Seattle, WA, USA, Workshop Proceedings*, 2011, pp. 63–68.