Markov-modulated Marked Poisson Processes for Check-in Data

Jiangwei Pan[1] PANJIANGWEI@GMAIL.COM
Vinayak Rao[2] VARAO@PURDUE.EDU
Pankaj K. Agarwal[1] PANKAJ@CS.DUKE.EDU
Alan E. Gelfand[3] ALAN@STAT.DUKE.EDU
[1] Dept. of Comp. Sci., Duke University, [2] Dept. of Statistics, Purdue University, [3] Dept. of Statistical Sci., Duke Univ.

Abstract

We develop continuous-time probabilistic models to study trajectory data consisting of times and locations of user 'check-ins'. We model the data as realizations of a marked point process, with intensity and mark-distribution modulated by a latent Markov jump process (MJP). We also include user-heterogeneity in our model by assigning each user a vector of 'preferred locations'. Our model extends latent Dirichlet allocation by dropping the bag-of-words assumption and operating in continuous time. We show how an appropriate choice of priors allows efficient posterior inference. Our experiments demonstrate the usefulness of our approach by comparing with various baselines on a variety of tasks.

1. Introduction

With the advancement of sensing technologies, trajectory data are being collected in scenarios such as traffic monitoring (Wang et al., 2014; Westgate et al., 2013), surveillance systems (Wang et al., 2011), and mobile and social network check-in data (Zhang et al., 2014; Gao et al., 2012a). A trajectory is a path taken by an object over a time period, with data consisting of observations at a discrete, often irregular, sequence of times. In the simplest case, each observation consists of a time and a location (latitude and longitude); it can also include activity and user information. The observations may be noisy, due to measurement error or human perturbation. In many cases, the trajectory sample points are sparse, with large gaps between bursts of observations. Furthermore, the rate of observations can vary along the trajectory, and can depend on the the user and the location of the trajectory.

Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48. Copyright 2016 by the author(s).

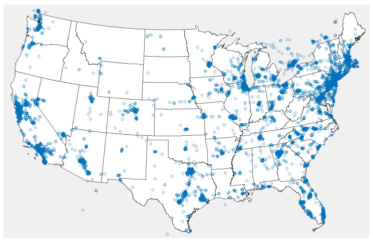


Figure 1. Visualizing the check-in locations (in the US) of 500 random users from the FourSquare dataset (Gao et al., 2012a).

Our focus in this paper is on user check-in data. In particular, we consider a dataset of FourSquare check-ins (Gao et al., 2012a) (see Figure 1). This smartphone-based social media tool allows individuals to publicly register visits to interesting places: a check-in is a record of such a visit, consisting typically of a timestamp, a location (latitude and longitude), and additional information such as the name of a place, a category (e.g. restaurant or museum) or a user-rating. Unlike traditional trajectory data, check-in observations are not passively collected (say, every hour); instead, the times of observations are informative about user behavior. Consequently, observation times are irregular and typically alternate between bursty and sparse episodes. Our data is at the individual level, consisting of snapshots of individual trajectories over time intervals. The observations of each individual form a sequence of check-in times, each with an associated location.

Given a collection of trajectories from the same spatial domain, it is of interest to learn shared underlying patterns. For check-in data, such information can include activity hotspots (e.g. big cities, tourist attractions, restaurants), transition rates across different hotspots and nature of activity in different regions. The shared patterns can assist recommendation systems, managing traffic, identifying unusual activity patterns, understanding factors that lead to the success or failure of a business, as well as just to summarize a large collection of activity data.

We approach the problem of identifying shared patterns by proposing a realistic generative model for the check-in data and performing posterior inference for the model. We model check-ins as local excursions about a latent state: e.g., one might visit a restaurant in Chicago, Chicago being the latent state, and the restaurant coordinates giving the check-in location. Generatively, at the start of the observation window, the individual starts in a random state, stays there for a random period of time, checks in a random number of times while in that state, and stochastically transitions to a new state. When checking in, a random location is attached to the individual.

One possible generative model is a state-space model (Fuller, 1996). Discrete-time models of dynamical systems are widespread in machine learning and statistics: starting with the simple hidden Markov model (Rabiner, 1989), extensions like dynamic Bayesian networks (Murphy, 2002), infinite-state HMMs (Beal et al., 2001; Teh et al., 2006), factorial Markov models (Ghahramani & Jordan, 1997) have been proposed and applied. These models are however unsuitable for the check-in data we consider, which contains bursty, asynchronous observations over long intervals of time. As we demonstrate in our experiments, accurate prediction requires artificial discretization at high temporal resolution, incurring expensive computation.

In this work we apply ideas from continuous-time Markov jump processes (MJP) to model the check-in data. Unlike their more common discrete-time counterparts, such models are ideal for bursty, asynchronous observations. While MJPs have found applications in modeling point data (e.g. event attendance (Ihler et al., 2007), genetics (Fearnhead & Sherlock, 2006)), there is little work applying MJPs to spatio-temporal phenomena like check-in data. One of our contributions is to apply and extend MJPs to construct a marked Markov modulated Poisson process model for such data. Another contribution is to introduce heterogeneity across point processes via user-specific 'preference' vectors. The result can be regarded as an extension of latent Dirichlet allocation (LDA) (Blei et al., 2003) by dropping the bag-of-words assumption and operating in continuous time. We also introduce a convenient conjugate prior that allows efficient posterior inference by extending existing Markov chain Monte Carlo (MCMC) methods. We apply our model to the FourSquare check-in data, visualize the shared pattern inferred by the model, and demonstrate its advantage over HMM and LDA for data-exploration, prediction, and anomaly detection. More generally, our model is applicable to other types of time-stamped sequence data.

Related work: There has been much work on learning shared patterns of trajectories using model and optimization-based approaches. By treating trajectories as documents of exchangeable words, topic models such as latent Dirichlet allocation (LDA) (Blei et al., 2003), originally proposed to learn shared topics from documents, are well-suited for this task. Wang et al. (2011) extended the hierarchical Dirichlet Process, a nonparametric variant of LDA, for trajectories from videos and other surveillance devices. The topics learned by their model represent semantic regions of different activities. These work ignore sequential information in the trajectories. Zhang et al. (2014) approach the problem of extracting semantic sequential patterns from user check-in trajectories via a two-step optimization algorithm that uses category information of each check-in location; see also (Giannotti et al., 2007; Zheng et al., 2009) on discovering sequential patterns from trajectory data. Other shared patterns of trajectories that have been studied include *flock* (Laube & Imfeld, 2002), *convoy* (Jeung et al., 2008), and *swarm* (Li et al., 2010). Unlike the Bayesian approach that we take, these methods do not provide uncertainty in estimates, and are less capable of dealing with noise or missing data.

Various forms of discrete-time Markov models were used to model trajectory data. For example, Mathew et al. (2012) presented a hidden Markov model predicting the region of future locations based on previous visits to other locations. Gao et al. (2012b) considered both spatial and temporal information in their Markov model for location prediction. See also (Alvarez-Lozano et al., 2013; Xue et al., 2015).

Other related work includes Lichman & Smyth (2014), who applied a mixture of kernel density estimation approach on check-in data collected from Twitter and Gowalla for modeling individual-level location patterns, and Westgate et al. (2013), who used a Bayesian model on trajectories to estimate ambulance travel times.

2. Markov jump processes

A probabilistic mechanism well-suited for check-in data is the Markov jump process (MJP) (Çinlar, 1975). MJPs are the simplest processes for creating continuous-time extensions of a discrete-time Markov chain, and a realization is a piecewise constant function transitioning between N states. Changes in state can correspond to changes in location or activity. Such a continuous-time model allows us to effectively model the fact that typical trajectories make only few transitions over an extended period (e.g. a year), while accurately representing the actual transition times.

Formally, an MJP is parametrized by an initial distribution over states π, and an $N \times N$ rate matrix A. Entering state i triggers a race to jump to a new state $j \neq i$, with transitions from i to j having rate A_{ij}. Effectively, for each candidate new state j, a waiting time is sampled from an exponential distribution with rate A_{ij}. Suppose state j_0 has the minimum waiting time Δt_{j_0}. The system moves into state j_0 after Δt_{j_0} time. The process then repeats. Note that the off-diagonal elements A_{ij} are transition rates and are thus non-negative numbers. The negative of the diagonal element A_{ii} gives the rate of leaving state i (i.e. $A_{ii} = -\sum_{i \neq j} A_{ij}$) so that the rows of A sum to 0. An equivalent way of sam-

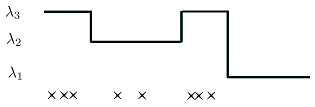


Figure 2. 3-state mMMPP with rates $(\lambda_1, \lambda_2, \lambda_3)$. Crosses are check-ins, having 2-d marks from a 2-d normal with state-dependent mean and covariance.

pling an MJP trajectory is to successively first sample the time of the next transition from an exponential distribution with rate $|A_{ii}|$, and then sample a new state $j \neq i$ with probability proportional to A_{ij}.

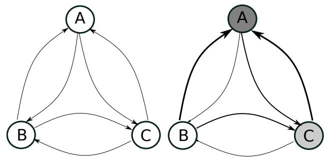
Modeling check-in data with MJPs: The data we consider consists of a collection of partially-observed trajectories over an interval $[0, T_{end}]$, each corresponding to a different 'user' (we will say users or trajectories depending on the context). We envision each trajectory as a piecewise-constant realization of an MJP. The piecewise constant function is characterized by the sequence of jump times T (including the start and end times 0 and T_{end}), and the value of the state at these times, S. We write this as $S(t) = (\mathsf{S}, \mathsf{T}) = ((s_0, t_0), (s_1, t_1), \ldots, (s_{|\mathsf{T}|}, t_{|\mathsf{T}|}))$, with $t_0 = 0$ and $t_{|\mathsf{T}|} = T_{end}$. We do not observe this sequence; instead, we have measurements at a finite random set of times (the user 'check-ins'). We allow the rate of check-ins to depend on the state of the MJP, and assign each state s a positive output rate $\lambda_s, s = 1, 2, ..., N$. Over intervals where the MJP is in state s, outputs are produced according to a Poisson process with rate λ_s. That is, the check-in times in state s are uniformly distributed over the time in state s. Altogether, the set of check-in times, H, forms a realization of an inhomogeneous Poisson process called a Markov-modulated Poisson process (MMPP).

Check-ins are characterized not just by time but also by location (which we call a 'mark'). With each state s, we associate a mean μ_s and a positive definite covariance matrix Σ_s, and model check-in locations as draws from a Gaussian with mean and covariance determined by the current state. The observations form a realization of a marked Poisson process, with an MJP-distributed intensity. We call this a marked Markov-modulated Poisson process (mMMPP); see Figure 2. While a location informs about which state an individual is in, it does not uniquely determine the state: in the space of locations, high-probability regions of states may not be geographically disjoint. The overall model is

$$\begin{aligned} S(t) &\sim \text{MJP}(\pi, A), \qquad (1)\\ H \equiv \{h_1, h_2, \ldots, h_{|H|}\} &\sim \text{PoissonProcess}(\lambda_{S(t)}),\\ \forall i \in \{1, \ldots, |H|\} \quad x_i &\sim \mathcal{N}(\mu_{S(h_i)}, \Sigma_{S(h_i)}). \end{aligned}$$

Here, the MJP state has two roles: it sets the rate at which the user produces check-ins as well as the distribution over locations. Usually this makes sense: different locations determine the rate at which the user produces check-ins. Sometimes, the check-in rate might just depend on the user,

Figure 3. Models without (left) and with (right) preference vectors. For the latter, a high preference for state A boosts incoming rates to A. A similar but smaller effect exists for C



or involve interaction between user and location.

We take a Bayesian approach, placing priors on the unknown π, A and λ. We place a Dirichlet prior on π and Gamma priors on the λ's and the off-diagonal elements of A. As the rows of A add to 0, the off-diagonal elements of A determines A, and we write this prior as $A \sim \text{oGamma}(\alpha_1, \alpha_2)$. We place a normal-inverse-Wishart prior (NiW) on the observation distribution parameters (μ, Σ) of each state. We show later that these priors are all conjugate and allow efficient Gibbs sampling.

User heterogeneity: The model above assumes trajectories are exchangeable across users (i.e., all trajectories are drawn from the same distribution). This is unrealistic; assuming states correspond to geographical regions, different users can favor different states (corresponding to where they live, work, or where their friends live). In a realistic scenario, the rate at which a user enters a state is determined by two factors: the preferences of the user and the popularity of that state. From the inference perspective, it is of interest to understand both global factors that drive transitions, as well as user-specific preferences. From the prediction perspective, it is important to allow individuals to deviate from global trends.

To capture both the global and user-level preferences, we propose a new prior over the rate matrix. We have N states, with a global rate-matrix A controling the rate of transitions across states. Every user u additionally has an N-dimensional *preference* vector B_u of nonnegative elements, giving user-specific state preferences. For this user, the rate of transitioning from state i to j is $A_{ij}B_{uj}$, the product of the global rate of tranition with the user's preference of state j (see Figure 3). This defines the off-diagonal elements of the effective rate matrix $\tilde{A}$, i.e., $\tilde{A}_{ij} = A_{ij}B_{uj}$; again, the diagonal elements are set so that the rows add up to 0. We place Gamma priors on the user-preferences B_{uj} and the transition rates A_{ij}. We call this MJP model with user preferences the *preference-MJP* model.

Our model relates closely to latent Dirichlet allocation (LDA), or topic, models. We can regard each individual check-in sequence as a document, and the state-specific distributions as 'topics'. Just as each document has its own distribution over topics, now, each individual has a vector of state preferences. Unlike LDA which assumes a document to be a bag-of-words, we model both the sequential

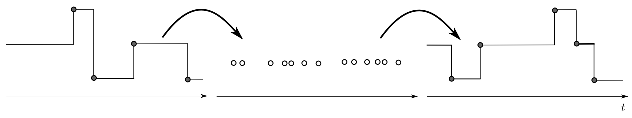


Figure 4. MCMC inference for MJPs

and temporal nature of check-in data. Note that the user preference vector in our model is unnormalized, and thus encodes not only user preferences but also transition rates.

3. Inference

We carry out inference via Markov chain Monte Carlo (MCMC), repeating two Gibbs steps: sample user trajectories given current model parameters, and then update parameters given trajectories. For the first step, we follow ideas in (Rao & Teh, 2013; 2012), sampling a new trajectory by running the forward-filtering backward-sampling algorithm (FFBS) (Frühwirth-Schnatter, 1994) on a *random* discretization of time. In every Gibbs iteration, this discretization is resampled conditional on the old trajectory and the current parameters. More precisely, for some $\Omega > \max(-A_{ii})$, sample a set of times from a Poisson process with piecewise-constant intensity $\Omega + A_{s(t)s(t)}$. These times, along with the transition times of the old trajectory form a random discretization of time (the middle of Figure 4). A new trajectory is sampled by running FFBS over these times, with the Markov chain having transition matrix $(I + \frac{1}{\Omega}A)$ and initial distribution π (the right of Figure 4). Note that for MJP with user preferences, the matrix A above is replaced by the effective rate matrix $\tilde{A}$.

Conditioned on a discretization of time, the MJP state remains constant between two successive candidate times, with all observations lying in that interval determining the likelihood of that state. In our case, given observations (H^Δ, X^Δ) in an interval of length Δ, state s has likelihood $\left(\lambda_s^{|H^\Delta|}\exp(-\lambda_s\Delta)\right)\left(\prod_{i=1}^{|X^\Delta|}\mathcal{N}(x_i^\Delta|\mu_s,\Sigma_s)\right)$. The first term corresponds to the number of check-ins, and the second to their locations. Overall, sampling trajectories is simple and efficient, and while we have adapted it to the marked Poisson case, we refer the reader to (Rao & Teh, 2013) for details. We highlight that our inference algorithm does not require inefficient rejection sampling algorithms or expensive matrix exponentiation computations, unlike work such as (Hobolth & Stone, 2009; Bladt & Sørensen, 2005; Fearnhead & Sherlock, 2006; Metzner et al., 2007).

Inference over parameters and other latent variables also turns out to be straightforward for our choice of priors. For an MJP trajectory $S(t) = (\mathsf{S}, \mathsf{T})$ define $\Delta_i = (t_{i+1} - t_i)$ as the ith waiting time. Then $S(t)$ has likelihood

$$p(S(t)) = \pi_{s_0}\left(\prod_{i=0}^{|\mathsf{T}|-2}|A_{s_is_i}|\exp(-|A_{s_is_i}|\Delta_i)\cdot\frac{A_{s_is_{i+1}}}{|A_{s_is_i}|}\right)\exp(-|A_{s_{|\mathsf{T}|-1}s_{|\mathsf{T}|-1}}|(\Delta_{|\mathsf{T}|-1})) \quad (2)$$

Noting that $|A_{ii}| = \sum_{j\neq i}A_{ij}$, we then have

$$p(S(t)) = \pi_{s_0}\left(\prod_{i=0}^{|\mathsf{T}|-1}\prod_{j\neq s_i}\exp(-A_{s_ij}\Delta_i)\right)\prod_{i=0}^{|\mathsf{T}|-2}A_{s_is_{i+1}}$$

Letting W_i be the total amount of time spent in state i, and n_{ij} the number of transitions from i to j,

$$p(S(t)) = \pi_{s_0}\prod_{i=1}^{N}\prod_{j\neq i}\exp(-A_{ij}W_i)\cdot(A_{ij})^{n_{ij}} \quad (3)$$

This likelihood is conjugate to Gamma priors on the A_{ij}, making posterior sampling straightforward. Note that our prior is slightly different to that from (Rao & Teh, 2013) (who used a Gamma prior on the diagonal of A and unrelated Dirichlet priors on transition probabilities). The advantage of our prior is that it also allows easy sampling for the MJP model with user preference vectors. Recall that for this model, the A_{ij}'s are replaced by the product $A_{ij}B_j$. The likelihood now becomes $p(S(t)) = \pi_{s_0}\prod_{i=1}^{N}\prod_{j\neq i}\exp(-B_jA_{ij}W_i)\cdot(B_jA_{ij})^{n_{ij}}$. Now, a Gamma prior on A_{ij} is *conditionally* conjugate given B_j (and similarly for B_j given A). This results in a simple and efficient Gibbs sampler: 1) given the global rate matrix A and the preference vector B, calculate the effective rate matrix, and sample a new trajectory following (Rao & Teh, 2013), 2) given the trajectories and preference vectors of each user, sample the Gamma-distributed elements of A, and 3) given A and the user trajectories, sample the Gamma-distributed user preference vectors.

4. Experiments

We use a dataset of FourSquare check-in sequences from year 2011. Each check-in has a location (latitude and longitude) and a time stamp. The dataset was originally collected by (Gao et al., 2012a) to study location-based social networks. It has 8967 users, each having 191 check-in records on average. We only consider check-ins inside a rectangle containing the United States and parts of Mexico and Canada (see Figure 1). There is not much prior statistical modeling specifically for check-in data: one model we compare with is a variant of latent Dirichlet allocation (LDA) (Blei et al., 2003). Here, each check-in location is a word, each check-in sequence is a document, and a topic is represented as a Gaussian distribution over check-in locations. The LDA topics correspond to the states in our MJP models, and user topic-distributions give the personal preference of each user. LDA does not model sequential information and observation rates of the check-in sequences.

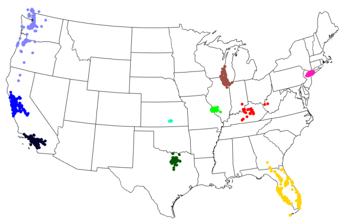


Figure 5. Visualization of representative MJP states.

current state	next states - probability
Bay Area	Napa-Sacramento - 0.30; LA - 0.19
Florida	New York City - 0.19; Chicago - 0.15
Los Angeles	Bay Area - 0.54; New York City - 0.15

Table 1. Most likely next states learned from transition rate matrix A of the preference-MJP model. States are manually labeled.

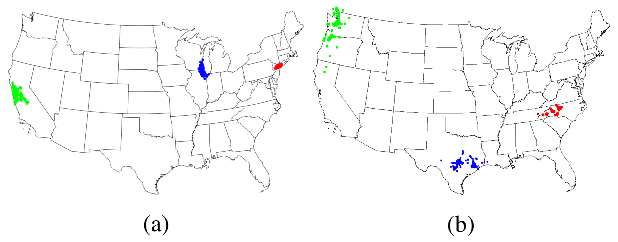


Figure 6. (a) Most attractive states (NYC, Chicago, San Francisco); (b) Most stable states (Charlotte, New Orleans, Seattle).

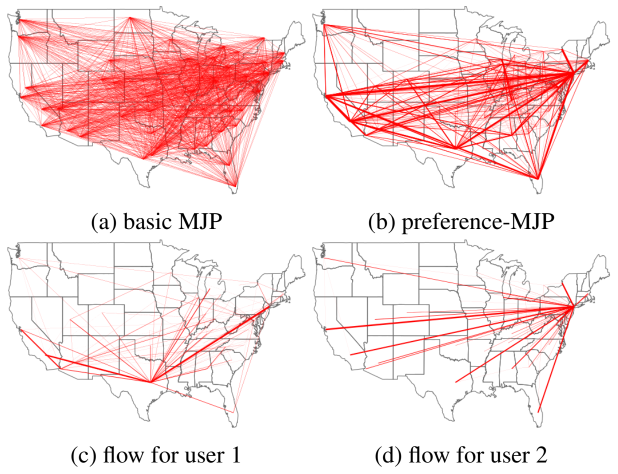


Figure 7. Flow between MJP states. Top row shows the global rate matrix A for (a) MJP and (b) preference-MJP. Bottom row shows user-specific flows for 2 users. Linewidth indicates flow.

	top preferences (rate/probability)
user 1	Seattle (7.92/0.207); Michigan (5.11/0.134)
user 2	NYC (11.06/0.186); Seattle (4.64/0.098)

Table 2. Top preferences of two users learned by the preference-MJP model. The numbers in brackets are corresponding values in the user's preference vector/normalized preference vector.

We perform most of our experiments using data at the national scale, but also look at a different granularity: check-ins only from Florida. In our experiments, we set the number of states and topics to 50 for the national data and 30 for the Florida data. In the national-scale MJP models, Florida corresponds to a single MJP state, while in the Florida-scale MJP models, each major city in Florida corresponds to a state. We fit all models using data from 500 random users, running the MCMC samplers for 1500 iterations and retaining the last 500 samples for the evaluation experiments described below. We use the preference-MJP model for the data visualization and exploration experiments.

Visualization: We first visualize representative MJP states on the US map, plotting all check-ins assigned to these states in Figure 5. Most states are local, corresponding to activity centers such as New York city (NYC) and Chicago. Information about the flow between states is contained in the rate matrix A, whose normalized rows give the probabilities of the next state. Table 1 gives a few examples. The diagonal entries of A (the sum of the off-diagonal elements in each row) show how soon users leave each state, or how long a typical visit to a state is. The total incoming rate of each state (sum of off-diagonal entries in a *column* of A) is a good measure of the "attractiveness" of the state. Figure 6(a) shows the top 3 "attractive" states in the US. Again, these correspond to NYC, Chicago, and the Bay area. Figure 6(b) shows the top 3 "stable" states, which have the longest typical visit and correspond to the rows of A with the minimum sums of off-diagonal entries. These states roughly are Charlotte, New Orleans and Seattle.

Figure 7(top row) shows the population flow according to the rate matrix A for (a) the basic MJP and (b) preference-MJP. For both, we have emphasized only large rates. The basic MJP has a denser rate matrix, connecting all urban centers in the US. By contrast, the global rate matrix for the preference-MJP model is much more sparse, since the preference vector of each user can incorporate user-specific transitions that might have low probability globally. Figure 7(bottom row) plots the effective transition rates from two users, which are quite sparse. Table 2 shows the top preferences of two more random users. Note that the user preference vectors learned by the our model highlight the "extra" preferences of users that deviate from the global trend. Such user-specific information allows better prediction about future activity than the basic MJP model. We investigate this in the next subsection. Finally, we visualize observation rates of states in the preference-MJP model in Figure 8. We take the last MCMC sample of the preference-MJP model parameters, and plot the observations assigned to each state in the same color from a spectrum of colors, where high rates correspond to red and low rates correspond to blue. As expected, large cities and tourism hotspots have higher rates of check-ins.

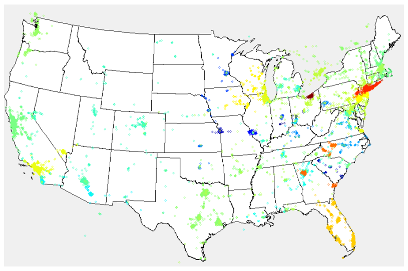


Figure 8. Different observation rates of the preference-MJP states. Rates increase from blue to red.

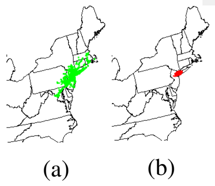


Figure 9. Observations assigned to the LDA topic (a) and the preference-MJP state (b) covering NYC.

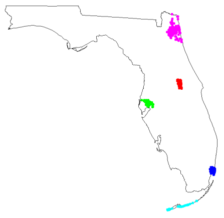


Figure 10. MJP states of the preference-MJP model on Florida check-in data. Roughly corresponds to Jacksonville, Orlando, Tampa, Miami and Key West.

For LDA, we observe that many of the 50 topics are "empty" and have few associated observations. The Gaussians of the remaining topics have larger variances compared to preference-MJP. For example, Figure 9 visualizes the LDA topic and preference-MJP state covering NYC. Clearly, the preference-MJP state is more fine-grained than the LDA topic, leading to better interpretation and prediction. Indeed, our preference-MJP model takes advantage of the sequentiality and check-in rate information embedded in the data while LDA only uses spatial information. The user topic distribution in LDA favors as few topics to explain each check-in sequence as possible, while preference-MJP can afford more states per check-in sequence as long as the transitions between them have high probabilities.

For the preference-MJP model trained on only Florida check-ins, the MJP states correspond to major urban centers such as Orlando, Miami, Key West, etc. See Figure 10 for a visualization of popular states. Figure 11 visualizes the observation rates of states in the same way as the national case. The rate matrix A also embeds interesting information. For example, the highest-rate destination from Orlando is Tampa, and the highest incoming rates of Key West are from Miami, the Northwest coast of Tampa and the area around Lake Jackson. See Figure 12 for the global rate matrix and the flow for a single user, whose major activities are near Tampa, Miami and Key West.

Prediction: To evaluate our model, we consider a binary classification task: given all check-ins of a user until a final check-in at time t, will the user make a new check-in in some region in the time interval $[t', t' + \Delta t]$ for some $t' \geq t$? We call $t' - t$ the *prediction gap* and $[t', t' + \Delta t]$ the *prediction interval*. In our experiments, we set $\Delta t = 0.01$

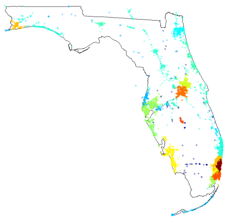


Figure 11. Different observation rates of the preference-MJP states in Florida. Rates increase from blue to red.

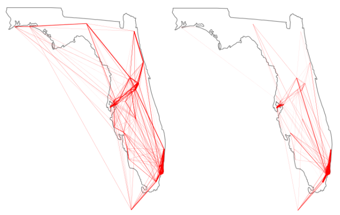


Figure 12. Global rate matrix (left) and flow for a user (right) for the Florida dataset with the preference-MJP.

year, and vary the prediction gap $t' - t$. Our prediction region is a rectangle $[40.544, 40.944] \times [-74.033, -73.833]$ that roughly covers New York city.

To construct a test case, we first select a user and randomly pick an index i in the second half of the user's check-in sequence. We assume the first i check-ins of this user are known. To make the prediction task harder, we choose test users from those whose check-in sequences have sufficient variance (at least a quarter of all-data variance). We create three test datasets with prediction gaps 0, 1/365, 0.1 years, each with 100 random test users with 50 positive cases (check-in in the prediction region during the prediction interval). Each positive test case is selected by first choosing a random user and then choosing a random index i in the second half of the user's check-in sequence such that the prediction interval contains a check-in in the prediction region (if it exists); else this will be a negative.

To make predictions using the preference-MJP model, we use the last 500 samples of the MCMC sampler. For each test user, we run the sampler on the known portion observations to obtain distributions over user-specific variables. We then simulate the model forward to make predictions. For every test user, we perform 500 iterations of Gibbs sampling of the user's hidden trajectory and preference vector, and use the last 200 samples for prediction. For each of the 200 trajectory samples, we record a positive if any extrapolated check-in within the prediction interval lands in the prediction region. The prediction score of a test user is the fraction of positive samples. Similarly, for LDA, we first estimate the number of observations in the prediction interval as $n_I = n\Delta t/t$, n being the number of check-ins of the test user in the interval $[0, t]$. We then estimate the topic preference distribution of the test user using observations up to time t and simulate n_I observations using the generative procedure of LDA. We record a positive if any of the n_I observations lies in the prediction region, and the prediction score is the fraction of positive records.

We also compare with a heuristic prediction method, which we call the *history* method. Given a test user, let n_α denote

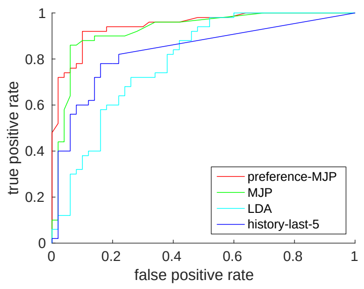


Figure 13. Comparison of ROC curves of the prediction methods with the gap 0 test data set.

method\prediction gap	0	1 day	.1 year
preference-MJP	**0.947**	**0.876**	**0.822**
MJP	0.927	0.870	0.713
LDA	0.794	0.782	0.759
history-all	0.838	0.801	0.767
history-last-1	0.810	0.790	0.710
history-last-5	0.835	0.813	0.792

Table 3. AUCs of MJP models and history heuristics on predicting if a user will check-in in NYC for different prediction gaps.

the number of check-ins that falls in the prediction region in time interval $(t-\alpha, t]$, where t is the last known observation time. The history method then computes the score of a test user as $n_\alpha \Delta t/\alpha$. We consider three settings, one with $\alpha = t$, namely, considering the whole known history of the user (history-all); one with $\alpha \to 0$, namely, considering only the last known check-in (history-last-1); and lastly, α such that the last 5 known check-ins are considered (history-last-5).

Figure 13 shows the receiver operating characteristic (ROC) curves of our models for the 0-prediction-gap task. For clarity, we only plot the curves for the MJP models, LDA and history-last-5; the area under curve (AUCs) of all methods are shown in Table 3. In all cases, the preference-MJP model performs the best. The advantage of the MJP models is most significant when the prediction gap is small because it exploits temporal information via the Markovian structure of the trajectories. As the prediction gap increases, the performance of the MJP model deteriorates, essentially using just the last check-in to predict the future check-ins. By contrast, the preference-MJP model uses the users' known check-ins to learn their preference vectors, which combined with global information from the training data makes more accurate predictions. The LDA model ignores time and sequential information in the data, harming performance. Of the history heuristics, the history-last-5 performs best, striking a balance between the other two extremes. Interestingly, although LDA uses user-level history and global spatial information, it does worse than the history-all heuristic, which only considers user-level history. This indicates that LDA topics may have oversmoothed spatial patterns of individual users.

The above task may be unfair to LDA which does not model observation rates. Thus we also compare preference-MJP and LDA on a variant of this task: instead of predicting whether a new check-in will be made in NYC in the prediction interval, predict the probability of a check-in in NYC in a given time interval conditioned on there being a check-in in that interval. We use the fraction of simulated check-ins in the prediction region as the prediction score. This prediction task is more difficult than before, but our preference-MJP model still outperforms LDA. Table 4 shows the average prediction errors of the two models.

method\prediction gap	0	1 day	.1 year
preference-MJP	**0.190**	**0.168**	**0.253**
LDA	0.252	0.208	0.291

Table 4. Absolute errors for predicting the probability of check-in in NYC with the prediction interval being .01 year and the prediction gaps being 0, 1 day or .1 year.

method\prediction gap	0	1 day	.1 year
preference-MJP	**0.885**	**0.829**	**0.801**
LDA	0.840	0.806	0.756
history-all	0.851	0.782	0.773
history-last-1	0.670	0.720	0.680
history-last-5	0.772	0.766	0.750

Table 5. AUCs predicting check-ins in Orlando from Florida data.

Finally, we repeat the binary version of the prediction task for the Florida data, with the prediction region as downtown Orlando, and with the same prediction gaps. Table 5 shows the various AUCs, and again, our model does best.

Anomaly detection: Anomaly detection involves detecting unusual user behaviour, and is important to detect fraud and identity theft. Anomalous behaviour might not involve obvious changes like a different distribution over locations or check-in rates, instead involving more subtle changes in patterns, e.g. a user choosing a different route to get to a location. Here, we use 100 random check-in sequences from the FourSquare dataset, of which 10 are anomalous. We obtain an anomalous sequence by randomly permuting locations of the second half check-ins of a random sequence, with the time stamps unchanged and sorted.

Given a check-in sequence, we obtain 200 posterior samples of the preference vectors from the first half and the second half. These samples are averaged to get the mean preference vectors for the two halves. Then the anomaly score of the sequence is set to the Euclidean distance between the two preference vectors normalized by the L_2 norm of the preference vector of the first half. The intuition for this is that the preference vector captures information about time spent at each state. On the other hand, the preference distributions over topics in the LDA model is not useful for this task since LDA treats check-ins as bag-of-words.

classifier	AUC	classifier	AUC
preference-MJP	**0.864**	grid-frequency	0.543
grid-wait-time	0.688	average-jump	0.713

Table 6. AUCs of classifiers for anomaly detection.

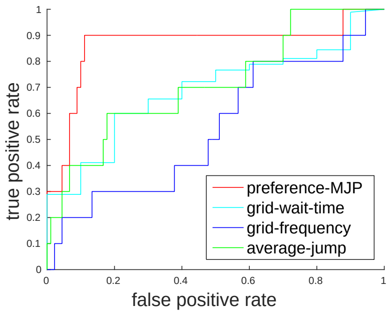


Figure 14. ROC curves of the preference-MJP based classifier and a histogram based classifier (with 50×50 grid) for detecting anomalous trajectories.

The only model for anomaly detection we are aware of is from (Ihler et al., 2007), though it was not designed for our choice of anomaly. We implement a variation of their model as follows: construct a uniform grid over the spatial domain and count check-ins in each grid cell for the two halves of the sequence. Then, compute the anomaly score as the Euclidean distance between the normalized frequency vectors of the two halves. We call this *grid-frequency*; Ihler et al. (2007) use a kernel density estimate instead. We also use two heuristic baselines. The first (called *grid-wait-time*) also constructs a uniform grid over the spatial domain, computes the average waiting time between consecutive check-ins in each grid cell, and computes the Euclidean distance between the grid average waiting time vectors of the two halves. The final baseline (called *average-jump*) computes the average distance between two consecutive check-ins for each half of the sequence. The anomaly score is the normalized difference between the two distances.

Figure 14 shows the ROC curves of our preference-MJP based classifier and the baselines (see Table 6 for AUCs). For the baselines, we tried different grid sizes and a 10×10 grid achieves the best AUC. Clearly, the preference-MJP based classifier outperforms the other baselines. This follows from using spatial, temporal, as well as sequential information in a coherent probabilistic way; each of the baselines uses just one of these pieces of information.

Discrete vs. continuous-time modeling: A simple alternative to the continuous-time MJP is the discrete-time hidden Markov model (HMM). This involves discretizing time into intervals of equal width δ, with the state remaining fixed over each interval, and evolving according to a Markov chain. Inference over the latent state given observations can be carried out with the usual forward-filtering backward-sampling algorithm that formed part of our MJP inference algorithm. A coarse discretization can result in check-ins over distant locations falling into the same interval. Since the latent state remains fixed over each time interval, a smaller discretization level models the data better. However, too fine a discretization can result in a long Markov chain and expensive computations. Our MJP model allows a few precisely located state-trasitions and strikes a good trade-off between efficiency and accuracy.

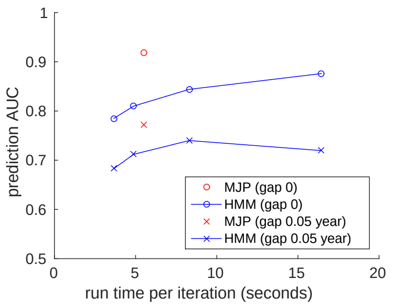


Figure 15. Trade-off between training time and prediction accuracy for 4 discretization levels in the HMM model.

We consider four different discretization levels $\delta \in \{0.1, 0.03, 0.01, 0.004\}$(year). For each, we divide the observation interval into T/δ time intervals, and estimate the state using FFBS. We place conjugate Dirichlet priors on the rows of the Markov transition matrix, allowing inference over the model parameters as well. For each discretization-level, and for the MJP, we evaluate model performance using our first prediction task . We also calculate the time required to run 1500 iterations of MCMC for training. Figure 15 shows the trade-off between training times and prediction accuracies (represented by AUC) of the HMM with the different discretization levels and the MJP, using the test datasets of prediction gaps 0 and 0.05 year. We can see that the MJP achives a good balance between training time and performance. This arises not only because it learns an effective granularity, but also because it allows this discretization-level to vary over the observation interval according to the observations.

5. Discussion

There are many interesting extensions to the work we presented here. While we fixed the number of states a priori, a natural extension learns this number by leveraging ideas from the nonparametric Bayes and Dirichlet process literature. This is related to extensions of models like (Saeedi & Bouchard-Côté, 2011) as well. There are other approaches to user heterogeneity that we have not included here: mixture models sharing rate matrices and mixture models that cluster preference vectors. It is also interesting to look at more flexible approaches, where user heterogeneity is determined by both the source and destination states. Finally, it is of interest to incorporate user-information (e.g., age, sex, occupation) as well as friendship networks that are typical of such datasets.

Acknowledgements

JP and PA were supported by NSF grants CCF-11-61359, IIS-14-08846, CCF-15-13816, ISS-14-47554, by ARO grant W911NF-15-1-0408, and by the U.S.-Israel Binational Science Foundation Grant 2012/229.

References

Alvarez-Lozano, Jorge, García-Macías, J Antonio, and Chávez, Edgar. Learning and user adaptation in location forecasting. In *Proceedings of ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication*, pp. 461–470, 2013.

Beal, Matthew J, Ghahramani, Zoubin, and Rasmussen, Carl E. The infinite hidden Markov model. In *Advances in Neural Information Processing Systems*, pp. 577–584, 2001.

Bladt, Mogens and Sørensen, Michael. Statistical inference for discretely observed markov jump processes. *Journal of the Royal Statistical Society: B*, 67(3):395–410, 2005.

Blei, David M, Ng, Andrew Y, and Jordan, Michael I. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

Çinlar, Erhan. *Introduction to Stochastic Processes*. Prentice Hall, 1975.

Fearnhead, Paul and Sherlock, Chris. An exact Gibbs sampler for the Markov-modulated Poisson process. *Journal of the Royal Statistical Society Series B*, 68(5):767–784, 2006.

Frühwirth-Schnatter, Sylvia. Data augmentation and dynamic linear models. *Journal of time series analysis*, 15 (2):183–202, 1994.

Fuller, Wayne A. *Introduction to Statistical Time Series*, volume 230. John Wiley & Sons, 1996.

Gao, Huiji, Tang, Jiliang, and Liu, Huan. gSCorr: Modeling geo-social correlations for new check-ins on location-based social networks. In *Proceedings of ACM International Conference on Information and Knowledge Management*, pp. 1582–1586. ACM, 2012a.

Gao, Huiji, Tang, Jiliang, and Liu, Huan. Mobile location prediction in spatio-temporal context. In *Nokia Mobile Data Challenge Workshop*, volume 41, pp. 44, 2012b.

Ghahramani, Zoubin and Jordan, Michael I. Factorial hidden markov models. *Machine Learning*, 29(2-3):245–273, 1997.

Giannotti, Fosca, Nanni, Mirco, Pinelli, Fabio, and Pedreschi, Dino. Trajectory pattern mining. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 330–339. ACM, 2007.

Hobolth, Asger and Stone, Eric A. Simulation from endpoint-conditioned, continuous-time markov chains on a finite state space, with applications to molecular evolution. *The Annals of Applied Statistics*, pp. 1204–1231, 2009.

Ihler, Alexander, Hutchins, Jon, and Smyth, Padhraic. Learning to detect events with markov-modulated poisson processes. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(3):13, 2007.

Jeung, Hoyoung, Yiu, Man Lung, Zhou, Xiaofang, Jensen, Christian S, and Shen, Heng Tao. Discovery of convoys in trajectory databases. *Proceedings of the VLDB Endowment*, 1(1):1068–1080, 2008.

Laube, Patrick and Imfeld, Stephan. Analyzing relative motion within groups oftrackable moving point objects. In *Geographic Information Science*, pp. 132–144. Springer, 2002.

Li, Zhenhui, Ding, Bolin, Han, Jiawei, and Kays, Roland. Swarm: Mining relaxed temporal moving object clusters. *Proceedings of the VLDB Endowment*, 3(1-2):723–734, 2010.

Lichman, Moshe and Smyth, Padhraic. Modeling human location data with mixtures of kernel densities. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 35–44. ACM, 2014.

Mathew, Wesley, Raposo, Ruben, and Martins, Bruno. Predicting future locations with hidden markov models. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pp. 911–918. ACM, 2012.

Metzner, Philipp, Horenko, Illia, and Schütte, Christof. Generator estimation of markov jump processes based on incomplete observations nonequidistant in time. *Physical Review E*, 76(6):066702, 2007.

Murphy, Kevin P. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UC Berkeley, 2002.

Rabiner, Lawrence R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–285, 1989.

Rao, Vinayak and Teh, Yee W. MCMC for continuous-time discrete-state systems. In *Advances in Neural Information Processing Systems*, pp. 701–709, 2012.

Rao, Vinayak and Teh, Yee W. Fast MCMC sampling for markov jump processes and extensions. *Journal of Machine Learning Research*, 14(1):3295–3320, 2013.

Saeedi, Ardavan and Bouchard-Côté, Alexandre. Priors over recurrent continuous time processes. In *Advances in Neural Information Processing Systems*, pp. 2052–2060, 2011.

Teh, Yee W, Jordan, Michael I, Beal, Matthew J, and Blei, David M. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

Wang, Xiaogang, Ma, Keng Teck, Ng, Gee-Wah, and Grimson, W Eric L. Trajectory analysis and semantic region modeling using nonparametric hierarchical bayesian models. *International Journal of Computer Vision*, 95(3):287–312, 2011.

Wang, Yilun, Zheng, Yu, and Xue, Yexiang. Travel time estimation of a path using sparse trajectories. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 25–34. ACM, 2014.

Westgate, Bradford S, Woodard, Dawn B, Matteson, David S, Henderson, Shane G, et al. Travel time estimation for ambulances using bayesian data augmentation. *The Annals of Applied Statistics*, 7(2):1139–1161, 2013.

Xue, Andy Yuan, Qi, Jianzhong, Xie, Xing, Zhang, Rui, Huang, Jin, and Li, Yuan. Solving the data sparsity problem in destination prediction. *The VLDB Journal*, 24(2): 219–243, 2015.

Zhang, Chao, Han, Jiawei, Shou, Lidan, Lu, Jiajun, and La Porta, Thomas. Splitter: Mining fine-grained sequential patterns in semantic trajectories. *Proceedings of the VLDB Endowment*, 7(9):769–780, 2014.

Zheng, Yu, Zhang, Lizhu, Xie, Xing, and Ma, Wei-Ying. Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of International Conference on World Wide Web*, pp. 791–800. ACM, 2009.