Predicting Unexpected Influxes of Players in EVE Online

Roman Garnett, Thomas Gärtner
University of Bonn & Fraunhofer IAIS
Bonn, Germany

Timothy Ellersiek
GIScience Research Group
University of Heidelberg
Heidelberg, Germany

Eyjólfur Guðmondsson, Pétur Óskarsson
CCP Games
Reykjavik, Iceland

***Abstract*—EVE Online is a massively multiplayer online role-playing game (MMORPG) taking place in a large galaxy consisting of about 7 500 star systems. In comparison to many other online role-playing games, the users interact in the same instance of a persistent player-driven universe. Given the number of simultaneous pilots online at the same time—a number which at times reaches up to more than 50 000 concurrent accounts logged on to the same server—the EVE Online universe can present atypically difficult load-balancing challenges when the users decide to operate in a coordinated fashion, for example, to launch an attack on a particular system. We will present an scalable, automated statistical method for predicting such unexpected user gatherings by considering the evolving shortest-path distances from each user to each system. Here we present a case study analyzing nearly 300 million user movements in the EVE Online universe from over 700 thousand user accounts over a period of three months. We demonstrate an ability to predict sudden spikes in user presence (corresponding to actual events) before they happen, suggesting our techniques could be useful for automated load-balancing in such massive online games.**

I. Introduction

EVE Online is a massively multiplayer online role-playing game (MMORPG), created by CCP Games, where thousands of users populate a single copy of an online universe. The game takes place in a galaxy with its star systems connected by "jump gates." The users play the roles of spaceship pilots that travel through the galaxy and interact with each other. Such interaction can take many different forms, including for example resource harvesting, trading, production of goods like spaceships, and fighting. In addition, the users can band together into corporations and alliances, wage war on each other, and assume control regions in space, central jump gates, or trading stations. The game has a vivid economy of produced goods and harvested resources, which is completely user-driven and not dictated by the game.

From time to time, user groups perform coordinated actions, such as a whole corporation harvesting asteroids to produce a large item together, or undertaking a surprise attack on another corporation. This unexpected and sudden gathering of several hundreds, or perhaps thousands, of users in a star system poses a huge load-balancing challenge to the servers hosting the spatial region of the game. CCP games hosts each star system on a specified set of colocated servers to reduce latency while processing inter-system events. If a sudden unexpected influx of users enters a system without warning, CCP Games might not have preallocated enough resources to that system to maintain a consistent user experience. For this reason, an automated method for predicting such gatherings before they occur could dramatically improve load-rebalancing decisions.

Here we present a case study analyzing nearly 300 million user movements in the EVE Online universe from over 700 thousand users over a period of three months. We demonstrate an ability to predict sudden spikes in user presence before they happen, suggesting our techniques could be useful for automated load-balancing in such massive online games.

The rest of the paper is arranged as follows. In the next section, we describe the data available to us while completing this case study. We then present a method for predicting the future path of a user in the EVE Online universe. Finally, we proceed to describe how these predictions can be aggregated to predict unexpected user gatherings. In both cases, we present experimental evidence of the efficacy of our proposed techniques.

II. Related Work

The field of spatial pattern mining considers the problem of analyzing patterns of spatially evolving patterns. Important applications of spatial/spatiotemporal mining include animal movement and transporation analysis, where the movements of groups of actors are inferred and used to find mobility patterns [1], [2] or to find regions of interest from a collection of trajectories [3], [4].

The widespread usage of geopositioning techologies (for example GPS) and as a consequence thereof vast amounts of mobility data—which can be used to pinpoint users to specific locations—have enabled an increased amount of research utilizing this spatiotemporal data to predict mobility patterns of people/objects in real-world applications. In [5], an approach is presented which calculates long term-predictions for end-to-end routes of vehicles based on GPS observations of the vehicles past trips. A selection of research in real-world scenarios to predict future locations can be found in [6], [7], [8], [9], [10], and [11]. Hereby the n most-recently visited locations are typically incorporated into a model (usually based on Markov chains) to make predictions for next location choices.

Ample research has also been conducted on virtual movement behaviour in games. For instance [12] and [13] feature approaches to identify and learn elementary movements from human players which are used to model game bots with human-like skills. In the context of game exploitation, [14] presents an approach to automatically detect the usage of game bots

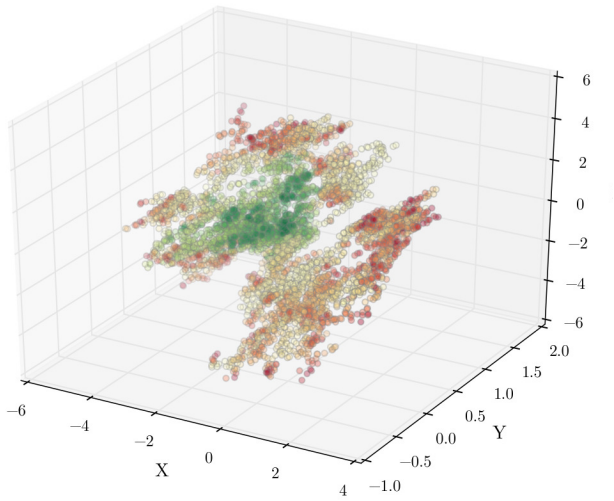

Fig. 1. The EVE Online universe, New Eden. The star systems are colored according to their security indices, from -1 (red) to 1 (green).

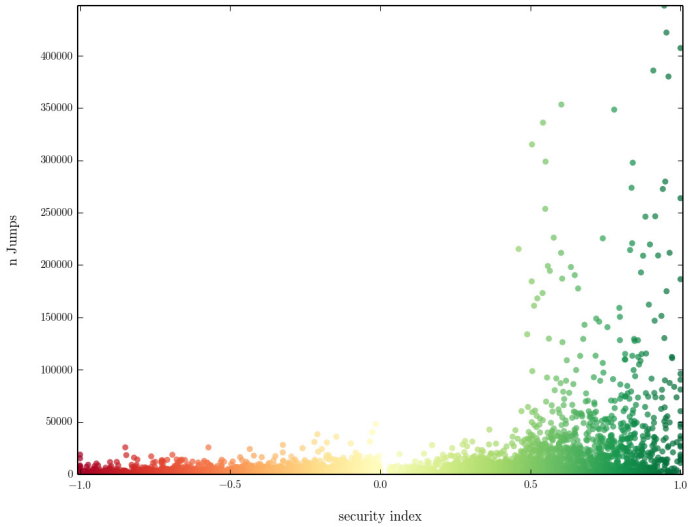

Fig. 2. Amount of jumps by the first 100 000 players grouped by all star systems and sorted by their security index, from -1 (red) to 1 (green).

in online games based on some dissimilarity measurements between the trajectories of either bots or human users.

Thus far, approaches to predict movement behaviour have typically been designed explicitly for two-dimensional spatial data collected from, e.g., the surface of the Earth. However, with the nowadays utterly sophisticated and complex online role-playing games, a further interesting field of research has emerged. Objective of our research is to introduce an approach to predict movement behaviour in a virtual reality — we thereby consider spatiotemporal trajectory mining in a complex jump-gate graph defined by the virtual New Eden universe of EVE Online.

III. Characteristics of New Eden and Description of Data

Before we describe the data we had available for us during this study, we pause to briefly describe the EVE Online universe. This presents unique challenges, which we overcome via a specially constructed (but quite general) probabilistic model.

A. New Eden

EVE Online takes place in a galaxy known as "New Eden." At the time this research was conducted, New Eden consisted of 5 201 permanent star systems and another 2 498 transitory systems located in a domain known as "wormhole space." Neighbouring systems are often connected with each other through a navigable network of "jump gates," which form the primary means of interstellar travel for players. Players can also purchase more advanced drive systems for faster-than-light travel avoiding this network. The bulk of activities and missions take place inside the star systems themselves, each of which spans a small portion of the overall space and can feature moons, planets, space stations, or even asteroid belts. Alliances of players can even build space stations of their own as outposts in certain insecure regions of New Eden. The bulk of New Eden, however, is partroled by non-player character (NPC) law enforcement units that preserve the balance inside central star systems, making them safe for travel, especially for those not necessarily equipped with advanced items.

The players are able to participate in various virtual professions, which can be trained by learning skills in real time. Bounty hunters, for instance, take on assignments to hunt down wanted players, logisticians generate virtual prosperity by moving valuable goods across the universe, and scientists apply their skills to the research and development of new materials manufactured out of resources. The spatial range of players in New Eden ultimately depends on which profession is chosen. One influencing factor in regards to which star systems are frequently visited is the *security factor,* which is grouped into three categories on a decimal scale between -1 and 1. Star systems with a high security have an index between 0.5 and 1.0 and are closely monitored by an artificial-intelligence–driven NPC police. Systems classified with a lower security, between 0.1 and 0.4, are comparativeley more dangerous as law enforcement units tend to be less rigorous towards aggressors. Finally, the so-called null security ("nullsec") space, which is entirely lawless, may be seized and reigned by powerful player alliances.

The shape of New Eden is elliptic-convex. The safest star systems can be found around the core of the universe, among which the most active and largest trading hubs are located. The farther away one moves from the core of the universe, the more unsafe it eventually becomes. Figure 1 illustrates the 5 201 star systems in normal space, colored according to their security index.

To provide some further insight into the characteristics regarding the spatial behaviour of the players, Figure 2 shows the number of arrivals (jumps) grouped by all star systems and sorted by their security indices. This points out clearly that the majority of jumps occur in the high security regions of the universe.

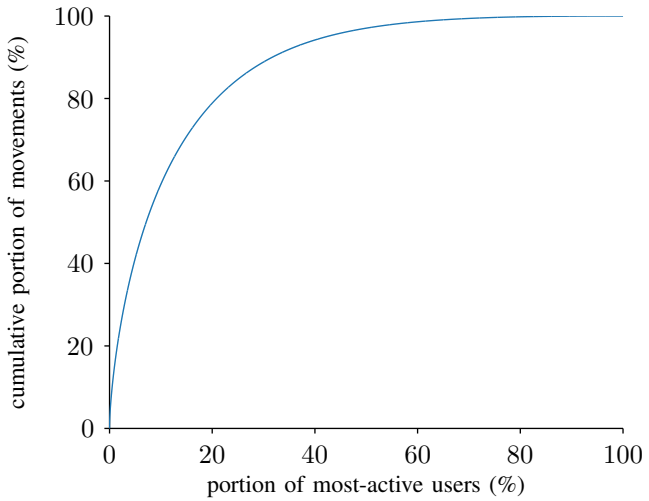

Fig. 3. Cumulative portion of movements represented by the most-active users.

B. Description of data

The data we had available for investigation consisted of the trajectories (space and time coordinates) of 720 835 unique users who jump between the 5 201 star systems, using a total of 14 334 jump-gates. As mentioned above, there are different ways to travel from one star system to another in the EVE Online universe (e.g., wormholes and so-called "clone-jumps"). For the purpose of our analysis, we solely consider regular jump-gate jumps that connect the systems in normal space.

CCP games provided us with user movement data from EVE Online gathered from 1 June to 13 September 2012. The format of the data was a list of records of the following form:

`[userid], [date], [time], [destination id], [type]`

In total, the dataset comprises 286 809 755 such records. A total of 720 835 unique user ids are seen in the dataset. The `[destination ids]` can represent systems in the EVE universe or space stations therein.

The mean number of total jumps made over all users is 397, the median is 84 and the comparably high standard deviation is approximately 897 jumps. The high variance is accounted for by the fact that the majority of the movements are caused by a minority of the users. Figure 3 shows the portion of all movements accounted for by the most-active users for any given threshold. Notice that 80% of all movements are generated by the 20% most active users, who averaged approximately five movements per day during the window of activity available to us.

Because the jump-gate graph is very sparsely connected (5 201 vertices, 14 334 edges) a user can usually not directly jump to his destination, but is forced to travel a route of intermediate systems. Figure 4 shows the shortest-hop distances between every pair of systems in normal space. One can see that traveling between two random systems most of the time takes somewhere between 20 and 60 jumps, which corresponds to a traveling time of roughly 20-70 minutes. This means that the users cannot congregate in a region immediately, but have to travel there, which usually takes some time. This delay of arrival motivates our research to predict the gatherings before they actually happen.

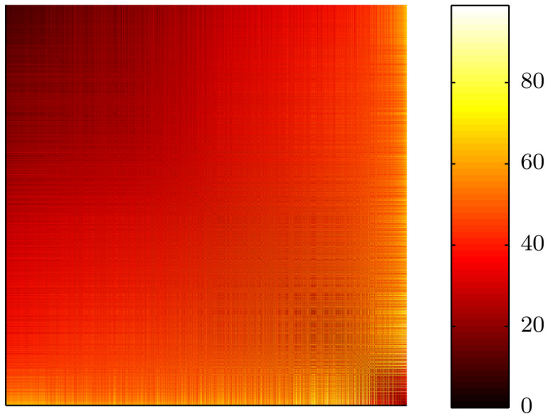

Fig. 4. The shortest-hop-distances from any star system to any other in the EVE Online universe, sorted by their row and column sum.

C. Preproccessing

For the purposes of this study, we applied mild preprocessing to the provided data. First, location ids corresponding to space stations were replaced with the location ids corresponding to the systems containing them. Second, we discarded movements into or out of "wormhole space" due to the transitory nature of the jump gates into and out of these systems. Less than 5% of the total movements were discarded by this filtering step. Finally, a small number of the user ids were equal to NULL; the corresponding movements were discarded as well. In total we were left with a little over 273 million movements among the 5 201 systems in normal space.

IV. Predicting User Movements

In this section we will consider the problem of predicting the next location(s) of users in the EVE universe, given their observed behavior. Due to the massive scale of the data involved, we sought a method that was both accurate and efficient to implement.

Our proposed method is probabilistic in nature. We assume a movement model wherein, at all times, a user has an unknown and never observed latent destination (here a system in the EVE universe), that he or she is trying to get to. After arriving at their destination, the user must then choose a new destination and proceed towards it. Conditioned on a given destination, we assume that the user mostly makes "efficient" moves, that is, they usually make movements that bring them closer to their destination. For mining the EVE movement data, we capture the notion of "closeness" via the shortest-path distance in the jump-gate graph between the user's current location and a latent destination.

A. Notation

Let us establish some notation. We will notate the normal-space jump graph as $G = (\mathcal{S}, E)$, where the nodes $\mathcal{S}$ comprise the set of systems in normal space and the set of (undirected) edges E correspond to jump gates. For two systems s, $s' \in \mathcal{S}$, we will write $d_{\text{SP}}(s, s')$ to indicate the shortest-path distance

between s and s' in G with unit edge weight. We will also write $\mathcal{N}(s) \subset \mathcal{S}$ to indicate the neighborhood of s. Finally, given systems $s, d \in \mathcal{S}$, we will write

$$\mathcal{C}(s,d) = \big\{s' \in \mathcal{N}(s) \mid d_{\text{SP}}(s',d) < d_{\text{SP}}(s,d)\big\},$$
$$\bar{\mathcal{C}}(s,d) = \big\{s' \in \mathcal{N}(s) \mid d_{\text{SP}}(s',d) \geq d_{\text{SP}}(s,d)\big\} = \mathcal{N}(s) \setminus \mathcal{C}(s,d),$$

for the set of nodes adjacent to s that are closer to a given destination d than s—$\mathcal{C}(s,d)$—and its complement.

B. Movement model and inference

Our model for how users move throughout the jump graph will be simple, generative in nature, and allow easy and fast inference. A high-level summary of the model is as follows. After entering a system, a user decides on a destination, which can be any other system in the universe. Given this destination, he or she chooses from among the neighboring systems where to move next. After moving there, a new destination is chosen, and we continue. We furthermore make the following simplifying (but quite reasonable) assumptions:

1) Users usually move towards their destinations.
2) Destinations are relatively stable throughout time (users are not excessively fickle).

Consider a user u. We will associate two discrete-time–indexed random variables associated with u, $s_i(u)$ and $d_i(u)$. The former represents the observed sequence of systems visited by the user, that is, $s_i(u)$ represents the ith system visited by u. The latter represents the unobserved destination chosen by u after arriving at system $s_i(u)$. We will henceforth drop the explicit reference to u for these random variables when there is no possibility of ambiguity.

Suppose, now, that we have observed u at system s_i. We assume that, given a chosen destination d_i, the user more often than not moves "towards" their destination, where closeness is captured by d_{SP}. We introduce a fixed parameter $\alpha < 1/2$ that encodes the probability that a user makes a movement that does not progress them towards their chosen destination. Assuming no other bias leads to the following conditional probability for the next location, s_{i+1}:

$$p(s_{i+1} \mid s_i, d_i) = \begin{cases} (1-\alpha)/\big|\mathcal{C}(s_i,d_i)\big| & s_{i+1} \in \mathcal{C}(s_i,d_i), \\ \alpha\ /\big|\bar{\mathcal{C}}(s_i,d_i)\big| & s_{i+1} \in \bar{\mathcal{C}}(s_i,d_i), \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

After choosing and arriving at s_{i+1}, the user u must then select a new destination d_{i+1}. To enable inference over the latent destinations, we must assume a model for this decision process as well. Here we make two assumptions. First, if the user arrives at what had been his or her destination, he or she must select a new (and different) destination. Second, if the user does not arrive at what had been his or her last destination, then they most likely remain on their chosen course. That is, users are not likely to change their minds "mid-course." We introduce a parameter $\beta < 1/2$ that represents the probability of a user selecting a new destination without having arrived at their previous destination. This leads us to the following:

$$p(d_{i+1} \mid s_{i+1}, d_i) = \begin{cases} q_{\setminus s_{i+1}}(d_{i+1}) & s_{i+1} = d_i \\ 1-\beta & s_{i+1} \neq d_i \wedge d_{i+1} = d_i \\ \beta q_{\setminus s_{i+1}}(d_{i+1}) & s_{i+1} \neq d_i \wedge d_{i+1} \neq d_i, \end{cases} \quad (2)$$

where $q(d)$ is an arbitrary and given prior distribution over destinations, and $q_{\setminus s}(d)$ represents the distribution resulting from setting $q(s) = 0$ and renormalizing. This convention disallows choosing the current location as a destination without having gone elsewhere first.

We have now fully specified our model for user movements. Unfortunately, the above probabilities are conditioned on the sequence of unknown latent destinations $\{d_i\}$. Fortunately, inference in this model is straightforward. We will describe the inference process for the latent destinations inductively. We begin with the simple base case where a new user has started playing the game in system s_1, without ever having been anywhere else. In this case, the user had no previous destination, and must simply start fresh:

$$p(d_1 \mid s_1) = q_{\setminus s_1}(d_1). \quad (3)$$

Now we describe the inductive step. Assume we have observed the user move through systems $S_i = (s_1, s_2, \ldots s_i)$ and have derived the distribution $p(d_i \mid S_i)$, and now the user moves from s_i to s_{i+1}. We describe how to calculate $p(d_{i+1} \mid S_{i+1})$. First, we update our belief about the previous destination, d_i, given this new information via Bayes' rule:

$$p(d_i \mid S_{i+1}) \propto p(s_{i+1} \mid s_i, d_i) p(d_i \mid S_i), \quad (4)$$

where both right-hand side distributions are known. Finally, we may calculate our belief over the new destination d_{i+1}. Let π represent the probability that s_{i+1} was the the user's last destination d_i (and that he or she is therefore compelled to choose a new destination):

$$\pi = \Pr(s_{i+1} = d_i \mid S_{i+1}). \quad (5)$$

Now applying (2), we have

$$p(d_{i+1} \mid S_{i+1}) = \big(\pi + \beta(1-\pi)\big) q_{\setminus s_{i+1}} + (1-\pi)(1-\beta) p(d_i \mid S_{i+1}). \quad (6)$$

We may continue in this manner until we have processed all of u's movements.

C. Predicting movements

In the previous subsection, we described how we may perform inference about the sequence of latent destinations given observations of a user's movements. We may also use our model to predict the next locations a user will visit.

In the simplest case, given the distribution $p(d_i \mid S_i)$, we may predict the next location, s_{i+1}, by marginalizing out the current unknown destination d_i:

$$p(s_{i+1} \mid S_i) = \sum_{d_i \in \mathcal{S}} p(s_{i+1} \mid s_i, d_i) p(d_i \mid S_i), \quad (7)$$

where $p(s_{i+1} \mid s_i, d_i)$ is given in (1).

We may also predict several steps into the future, but this is a little less straightforward. We consider as an example predicting the location two steps into the future, s_{i+2}. We have

$$p(s_{i+2} \mid S_i) = \sum_{s_{i+1} \in \mathcal{S}} p(s_{i+2} \mid S_i, s_{i+1}) p(s_{i+1} \mid S_i), \quad (8)$$

where now we have to condition our model on every possible intervening system s_{i+1}. This can quickly become expensive as the prediction horizon gets longer and longer. Consider the general case for predicting k steps in the future:

$$p(s_{i+k} \mid S_i) = \sum_{\substack{s_{i+j} \in \mathcal{S} \\ 1 \le j \le k-1}} p(s_{i+k} \mid S_i, \cup_{j=1}^{k-1} s_{i+j}) p(\cup_{j=1}^{k-1} s_{i+j} \mid S_i), \quad (9)$$

which requires a sum over $|\mathcal{S}|^{k-1}$ unknown variables. Here we propose two approaches to making longer-horizon predictions in a computationally feasible way.

The first suggestion is appropriate when we only want to make a prediction about future destinations and do not require a full probability distribution over the potential systems. Suppose we have chosen a prediction horizon k and want to predict the sequence $(s_{i+1}, s_{i+2}, \ldots, s_{i+k})$ given the sequence of past locations S_i. Rather than consider all possible sequences of length k, we simply perform k successive one-step predictions using (7), and assume the user moves to the most likely system at each step. To illustrate, let $\hat{s}_{i+1}$ denote the most likely next location for our user at the next time step:

$$\hat{s}_{i+1} = \arg\max_{s_{i+1} \in \mathcal{S}} p(s_{i+1} \mid S_i). \quad (10)$$

Naturally, we choose $\hat{s}_{i+1}$ as our prediction for s_{i+1}. Now we condition our model on the user moving to $\hat{s}_{i+1}$, and find the most likely next destination assuming $s_{i+1} = \hat{s}_{i+1}$:

$$\hat{s}_{i+2} = \arg\max_{s_{i+2} \in \mathcal{S}} p(s_{i+2} \mid S_i, s_{i+1} = \hat{s}_{i+1}). \quad (11)$$

We again condition on $s_{i+2} = \hat{s}_{i+2}$ and continue in this manner until we have found $\hat{s}_{i+k}$. We output the sequence $(\hat{s}_{i+1}, \hat{s}_{i+2}, \ldots \hat{s}_{i+k})$ as our prediction. Although this sequence is not guaranteed to be the most likely sequence if we considered all possible paths of length k, it requires conditioning the model only $k-1$ times, and therefore the computational expense grows linearly with the horizon. We will see later that this approach can predict the future locations of users with reasonable accuracy.

If we require a probability distribution over the location s_{i+k}, but want to avoid computing the full probability distribution, we may instead use Monte Carlo tree search. Here, we sample possible sequences by successively computing our belief about the next location, sampling a system from that distribution, and conditioning on the user moving to that distribution. We repeat this process several times, and output the empirical distribution over s_{i+k} seen in our samples as our estimate of the true probability. This procedure is guaranteed to converge to the true distribution as the number of samples increases.

D. The choice of prior

In the discussion above, we left the choice of prior distribution $q(d)$ open. In general this is arbitrary, but we can suggest several reasonable possibilities here.

The simplest choice is to use a uniform prior over destinations; however, we should not expect this to work very well. In the EVE universe, some systems are visited much more often than others; for example, the top-1% most-visited destinations account for almost 25% of all movements. A better choice would be to use an empirically chosen distribution from available data, for example, to find the empirical distribution of total arrivals or total time spent in each system.

Here we suggest a more sophisticated approach wherein we learn a personalized prior for each user based on their movements. First we choose a reasonable global prior from empirical data, $q_*(d)$, as in the above paragraph.[1]

Consider a user u. Before having seen any movements by u, we use the global prior $q_*(d)$. Suppose now that we have observed u make the sequence of moves $S_i = (s_1, s_2, \ldots s_i)$. Let c represent the $|\mathcal{S}|$-dimensional vector obtained by aggregating counts for each system in S_i (so that $\|c\|_1 = i$). We now use the following prior for u:

$$q(d) = \frac{\gamma q_*(d) + c}{\gamma + i}, \quad (12)$$

where $\gamma > 0$ is some chosen constant. We therefore use a mixture of the global prior with the empirical distribution over systems visited by u, with the parameter γ controlling how much emphasis we place on the global prior.[2] As the number of observations of u increases, we will weight the empirical distribution more and more heavily. This is the scheme we use for setting $q(d)$ throughout our experiments.

E. Movement prediction experiment

Before considering the problem of predicting unexpected gatherings, we pause to investigate the ability of our proposed model to predict individual user movements. We carried out a simple experiment to this end.

We began by extracting 10 000 random users with at least 500 total movements in our preprocessed dataset. We then compared the ability of our model and two baseline approaches to predict a given user's next 10 visited destinations. For all approaches, we used the sequential maximum-likelihood approach described above (10). The only difference between approaches was the choice of transition probability $p(s_{i+1} \mid S_i)$. Our model's choice is given in (7); the two baselines are:

1) RANDOM: transition probabilities are chosen uniformly at random,
2) PRIOR: the user always moves towards the most-visited system:

$$p(s_{i+1} \mid S_i) = \delta\big(\arg\max_{s \in \mathcal{N}(s_i)} q(s)\big). \quad (13)$$

For our model, we set the parameters as $\alpha = \beta = 0$; $\gamma = 1$.

[1] In our experiments, we set q_* by taking the histogram of $\min$(time spent in system s, 10 minutes) over the movements in the first week of data.

[2] This is equivalent to choosing a Dirichlet prior on the multinomial distribution $q(d)$ with parameter vector γq_*.

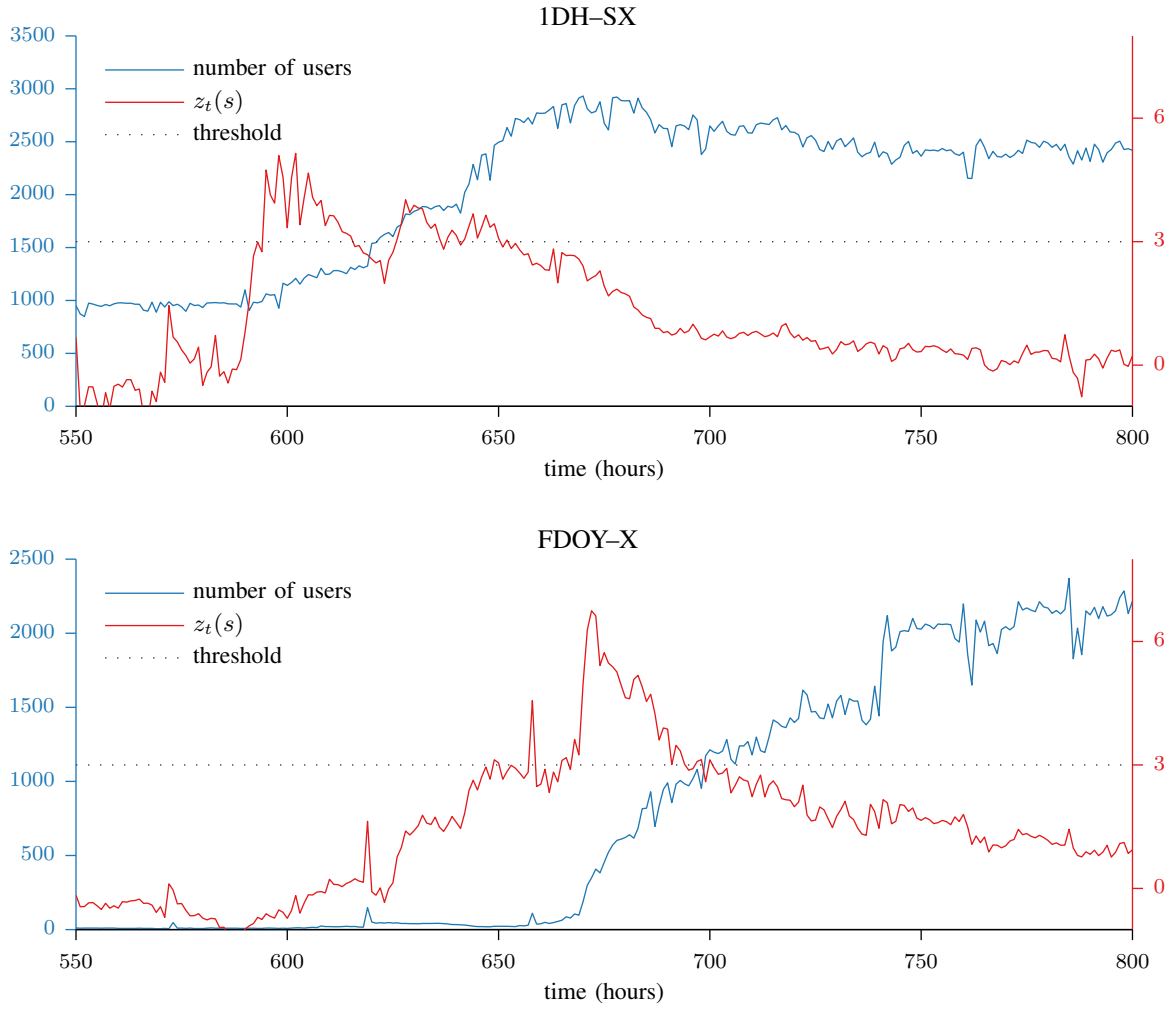

Fig. 7. An example of the gathering-detection procedure in progress for systems 1DH–SX (above) and FDOY–X (below). Both plots show the number of users present in the identified system (blue) vs. the alarm function $z_t(s)$ defined in the text (red). The horizontal axis is number of hours since midnight 1 June 2012; the plotted data correspond to approximately 24 June–4 July 2012. The sudden increase corresponds to the start of the 2012 Delve War.

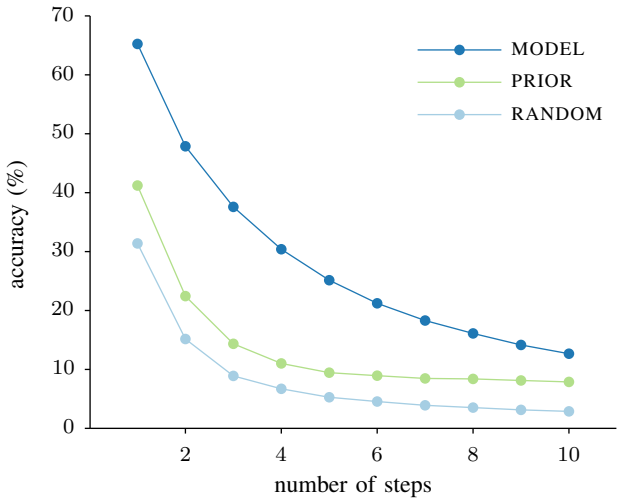

Fig. 5. Accuracy vs. number of steps ahead for the selected approaches in the prediction experiment.

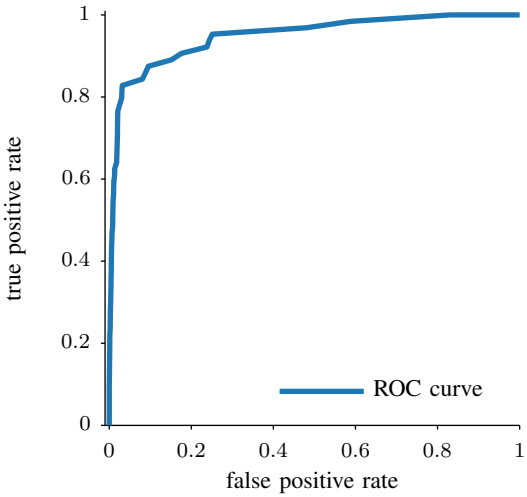

Fig. 6. ROC plot for the gathering-detection experiment.

For each of the chosen approaches and every prediction horizon $1 \leq k \leq 10$, we measured the overall predictive accuracy across all the available movements of the 10 000 selected users. The results are shown in Figure 5. It is readily apparent that the proposed model is much more accurate at predicting the future paths of users in the EVE universe than the chosen baselines. For this reason, we feel comfortable using it as the basis of our gathering-detection approach, described in the next section.

V. Predicting Gatherings

Now that we are equipped with a reasonably accurate method for analyzing and predicting the movements of players in the EVE Online universe, we proceed to describe how we can build upon it to predict large, unexpected gatherings of users before they happen.

Given the work in the previous section, our approach is relatively straightforward. We stream through the movements, keeping track of our probabilistic beliefs of each user's latent destination, using the model described above. Periodically (in our case, once an hour), we aggregate our beliefs about all currently active users. This gives us a probability distribution over all systems in normal space representing the probability that a user chosen uniformly at random has a particular system as his or her latent destination. To predict unexpected gatherings, we look for sudden upward movements in these values.

To be more precise, let us fix a particular and arbitrary system s. Suppose that $\{t_1, t_2, \ldots, t_n\}$ represents the set of times at which we chose to aggregate our beliefs (we assume that the time elapsed between each aggregation is constant). Let $p_t(s)$ represent the sequence of aggregated average probabilities that s is a given user's latent destination. We look for sudden increases in $p_t(s)$ by examining the evolution of simple statistics within a moving window of these values. Let w represent a chosen window width, and let t be an arbitrary time. We consider the values in the trailing window

$$W_t(s) = \big(p_{t-w}(s), p_{t-w+1}(s), \ldots, p_{t-1}(s)\big). \qquad (14)$$

Define $\mu_t(s)$ and $\sigma_t(s)$ to be the mean and standard deviation, respectively, of the probabilities in $W_t(s)$. After computing $p_t(s)$, we compute the z score of the current probability given the trailing window:

$$z_t(s) = \frac{p_t(s) - \mu_t(s)}{\sigma_t(s)}. \qquad (15)$$

Finally, we raise an alarm if $z_t(s) > \zeta$, for some chosen threshold ζ.

Figure 7 shows this procedure in action. We selected two systems, 1DH–SX and FDOY–X, that were involved in the so-called "2012 Delve War." Both systems experienced a sudden increase in the number of users present due to the onset of this extreme and unusual event—the latter system in particular never saw more than 38 total users present during the first three weeks of our data snapshot, but within a matter of hours, this number suddenly shot up to over 2 000. For both systems, we tracked the average destination probabilities of EVE users $p_t(s)$ on an hourly basis and calculated the $z_t(s)$ scores in (15) for a week-long trailing window. In both cases, the $z_t(s)$ function was predictive of the later increase in users, exceeding a typical alarm threshold of $\zeta = 3$ well in advance of the actual arrival of the involved users. This suggests that our proposed technique might have been useful for load-balancing or other purposes during this time period.

A. Gathering detection experiment

We implemented the simple gathering-detection mechanism above and conducted a simple experiment to investigate its ability to predict unexpected congregations before they occur. First, we scanned the available movement data to identify the sudden gatherings that do occur; here, we defined an "unusual gathering" as a time when a system experienced an average increase in presence of 200 users per hour, sustained over three hours. A total of 64 such gatherings occurred in our data, making them an exceedingly rare event.

We extracted all movements for reasonably active users in the dataset, defined as users that made at least 500 total movements over the 104 days in our snapshot. For each of these approximately 147 000 users, we used the model described in Section IV to process their movements and track our belief over their latent destinations. Again, the parameters were set as $\alpha = \beta = 0, \gamma = 1$. At hourly intervals, we aggregated the probabilities of all users seen thus far, producing the $p_t(s)$ signals for each of the 5 201 systems in normal space. We used a window of length one week to calculate the $z_t(s)$ scores described above. Finally, we varied the threshold ζ to produce a receiver–operator characteristic (ROC) plot, showing the number of true vs. false positive rates our proposed method can achieve on the given data. We considered our method successful for a given ζ if the $z_t(s)$ function would have exceeded ζ any time in the week preceding an unusual event in system s. Any other time $z_t(s)$ exceeded ζ was considered a false positive.

Figure 6 shows the results. The proposed method is able to detect over 80% of unusual gatherings with an exceptionally low false-positive rate. Note that many of these so-called "false positives" might indeed correspond to unusual gatherings that were less extreme than required by our quite stringent definition.

VI. Possible Routes of Improvement and Future Work

Although we consider our proposed methods quite successful already, we believe they could be improved with access to more data. Below we list several possible ways a practitioner with more access to information could improve upon this work.

- The EVE Online user interface allows a user to set an autopilot that will automatically guide them to a chosen system. If a user has set his or her autopilot to a particular system, we no longer need to perform inference over their latent destination d. Access to autopilot settings could improve the path-prediction model dramatically.
- We have thus far considered the likely destinations of each user completely independently. Given access to information regarding corporation/alliance membership, we could couple our beliefs of user behavior in potentially useful ways. We could, for example, analyze our destination beliefs on a corporation- or alliance-wise basis, allowing potentially weak signals to be amplified and become more obvious.

In addition to these, we believe there is interesting theoretical investigations that could lead to further success on related problems:

- We believe there is the potential to learn the transition probabilities $p(d_{i+1} \mid s_i, d_i)$ in a more sophisticated manner than we have done here. A model incorporating additional conditional information might do much better than the shared prior we chose here.
- The more general question of identifying groups of users who in some way act jointly could be both interesting and, if solved successfully, could possibly help us achieve even greater performance on this and related problems. We hope that using sophisticated clustering techniques, we might be able to discover hidden structure and patterns in the pilots' behavior.

REFERENCES

[1] Kalnis, P., Mamoulis, N., & Bakiras, S. (2005). On Discovering Moving Clusters in Spatio-temporal Data. In Proceedings of 9th International Conference on Advances in Spatial and Temporal Databases.

[2] Gudmundsson, J., Kreveld, M., & Speckmann, B. (2007). Efficient Detection of Patterns in 2d Trajectories of Moving Points. Geometrica, 11(2), 195-215.

[3] Giannotti, F., Nanni, M., Pinelli, F., & Pedreschi, D. (2007). Trajectory Pattern Mining. In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

[4] Masciari, E., Shi, G., & Zaniolo, C. (2013). Sequential Pattern Mining from Trajectory Data. In Proceedings of the 17th International Database Engineering & Applications Symposium.

[5] Froehlich, J., & Krumm, J. (2008). Route prediction from trip observations. SEA SP, 2193. 53.

[6] Gambs, S., Killijian, M. O., & del Prado Cortez, M. N. (2012). Next place prediction using mobility Markov chains. In Proceedings of the First Workshop on Measurement, Privacy, and Mobility (p. 3). ACM.

[7] Mathew, W., Raposo, R., & Martins, B. (2012, September). Predicting future locations with hidden markov models. In Proceedings of the 2012 ACM Conference on Ubiquitous Computing (pp. 911-918). ACM.

[8] Song, L., Kotz, D., Jain, R., & He, X. (2006). Evaluating next-cell predictors with extensive Wi-Fi mobility data. Mobile Computing, IEEE Transactions on, 5(12), 1633-1649.

[9] Gao, H., Tang, J., & Liu, H. (2012). Mobile location prediction in spatio-temporal context. In Nokia Mobile Data Challenge 2012 Workshop. p. Dedicated task (Vol. 2, No. 1). ISO 690.

[10] Ashbrook, D., & Starner, T. (2002). Learning significant locations and predicting user movement with GPS. In Wearable Computers, 2002.(ISWC 2002). Proceedings. Sixth International Symposium on (pp. 101-108). IEEE.

[11] Monreale, A., Pinelli, F., Trasarti, R., & Giannotti, F. (2009). WhereNext: a location predictor on trajectory pattern mining. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 637-646). ACM.

[12] Bauckhage, C., Thurau, C., & Sagerer, G. (2003) Learning human-like opponent behavior for interactive computer games. Pattern Recognition, (pp. 148-155).

[13] Thurau, C., Bauckhage, C., & Sagerer, G. (2004). Synthesizing movements for computer game characters. Pattern Recognition, LNCS 3175, (pp. 179-186).

[14] Pao, H., Chen, K., & Chang, H. (2010). Game bot detection via avatar trajectory analysis. Computational Intelligence and AI in Games, IEEE Transactions on, 2(3), (pp. 162-175).