Contextual Anomaly Detection Using Log-Linear Tensor Factorization

Alpa Jayesh Shah[1(✉)], Christian Desrosiers[1], and Robert Sabourin[2]

[1] Department of Software and IT Engineering, Ecole de technologie supérieure, Montreal, Canada
AShah@livia.etsmtl.ca, Christian.Desrosiers@etsmtl.ca
[2] Department of Automated Production Engineering, Ecole de technologie supérieure, Montreal, Canada
Robert.Sabourin@etsmtl.ca

Abstract. This paper presents a novel approach for the detection of contextual anomalies. This approach, based on log-linear tensor factorization, considers a stream of discrete events, each representing the co-occurence of contextual elements, and detects events with low-probability. A parametric model is used to learn the joint probability of contextual elements, in which the parameters are the factors of the event tensor. An efficient method, based on Nesterov's accelerated gradient ascent, is proposed to learn these parameters. The proposed approach is evaluated on the low-rank approximation of tensors, the prediction of future of events and the detection of events representing abnormal behaviors. Results show our method to outperform state of the art approaches for these problems.

Keywords: Contextual anomaly detection · Tensor factorization · Low-rank approximation · Future event prediction

1 Introduction

The recent commercialization of technologies for the real-time identification, location, and tracking of people and objects has opened the door to various new applications in domains such as safety, logistics and retail. Among these applications, the real-time detection of malicious or abnormal behaviors is of critical importance to the safety of the population.

The approaches proposed for this problem over the years can be roughly divided in two categories: the ones based on probabilistic generative models, and those using trajectory patterns. Approaches in the first category use a generative model, for instance based on Markov [7] or Hierarchical Dirichlet [9] processes, to determine the likelihood of a sequence of observed events/actions, and consider as abnormal behaviors the ones with a low probability. On the other hand, the second category of methods represent behaviors as trajectories through space, and considers as anomalies the trajectories that are significantly different from commonly observed ones. Trajectories can be encoded in various ways, such as sequences of points [16] or cubic splines [18]. Moreover, several approaches

T. Cao et al. (Eds.): PAKDD 2015, Part II, LNAI 9078, pp. 165–176, 2015.
DOI: 10.1007/978-3-319-18032-8_13

have been proposed to model the class of normal trajectories, for instance one-class SVM (OCSVM) [16], Gaussian Mixture Model [18], sparse coding [14] and frequent sub-sequence mining [10].

While these solutions are adequate in small and controlled environments, in which well defined activities occur, they usually perform poorly in large and dynamic environments, where the same sequence of events is almost never observed twice. In such complex environments, the context of events (such as location, duration, person ID, type of job, etc.) is often more important than their sequence. Thus, a person might take a slightly different route to go to the office, so analyzing the exact trajectory would likely result in many false positives. Moreover, even though the usual route is taken, this person's behavior can be abnormal if he/she goes to the office at an odd time (e.g., after 9 pm) or on a odd day (e.g., Sunday). This behavior might however be considered normal for other employees, such as security agents working on the evening or weekend shifts.

Although several approaches have been proposed to include contextual information in generative models, for example [3] and [5], these approaches are limited to specific contextual dimensions, such as the duration of events, and are unsuitable for complex environments. On the other hand, tensors have been recognized as a powerful and efficient method to model complex contextual information, and have been used in diverse applications like item recommendation [17] and analyzing email exchanges [1]. Recently, tensor factorization [4,12,21] has been explored as a novel way to detect anomalies, for instance, by tracking the reconstruction error over time [19], detecting outliers in the factor subspace [8,20], or decomposing a tensor as the sum of a low-rank component and a sparse residual representing the anomaly [11].

In this paper, we propose a novel method based on log-linear tensor factorization to detect contextual anomalies in large and complex environments. The advantages of this method are as follows:

1. Unlike existing tensor factorization approaches, which focus on detecting global anomalies [11,19] or distance-based outliers [8,20], our method learns the joint distribution of contextual dimensions. This allows it to evaluate the true probability of incoming events and mark low probability ones as abnormal. Our method also has the ability to detect specific types of anomalies efficiently, by using the probability of a given dimension conditioned over the other ones.
2. While most tensor factorization approaches are based on a linear model, our method uses a log-linear model, which can learn more complex relations in the data. Moreover, the proposed model implicitly enforces non-negativity in the tensor, a useful property when dealing with count data. In comparison, state of the art factorization techniques like Non-negative Tensor Factorization (NTF) [21] and Alternating Poisson Regression (APR) [4] impose non-negativity by constraining the factors, making the inference process more complex.
3. The proposed method uses an efficient inference strategy, based on Nesterov's accelerated gradient ascent, which has a complexity comparable to the state of the art Alternating Least Square (ALS) method [17], but offers more flexibility (e.g., ALS is limited to linear models and does not impose non-negativity).

The rest of this paper is divided as follows. In Section 2, we describe our proposed model, its inference strategy, and the method used to detect anomalies. Section 3 then evaluates our model on the tasks of approximating tensors using a small number of parameters, predicting the occurrence of future events and detecting abnormal events from their context. Finally, we summarize our contributions and results in Section 4.

2 The Proposed Method

2.1 Model Description

We model the multi-dimensional context of an event using a set of discrete random variables $\{X_1, \ldots, X_D\}$, representing the identifier (ID) of contextual elements like person, zone, time of day, etc. Each variable X_j has domain $\Omega_j = \{1, \ldots, N_j\}$, and we denote as $X_j = i_j$ the observation of element $i_j \in \Omega_j$ for dimension j. To simplify the notation, we use x_{i_j} as shorthand for this observation. For example, if the first dimension represents people, then x_{i_1} means the observation of person i_1, from a group of N_1 people, in the event.

We suppose that the observation of events depend on a set of latent factors $\mathcal{Z} = \{Z_1, \ldots, Z_D\}$, providing high-level information about the contextual elements. We define as $\boldsymbol{z}_{i_j} \in \mathbb{R}^K$ the latent factor vector corresponding to the i_j-th element of dimension j, where K is a user-supplied parameter. Using the previous example, $\boldsymbol{z}_{i_1}$ would be the latent factor vector of person i_1.

To model the joint probability of contextual elements, we use the following log-linear model:

$$p(x_{i_1}, \ldots, x_{i_D} \mid \mathcal{Z}) = \frac{\exp\left(\langle \boldsymbol{z}_{i_1}, \ldots, \boldsymbol{z}_{i_D} \rangle\right)}{\sum_{i'_1=1}^{N_1} \cdots \sum_{i'_D=1}^{N_D} \exp\left(\langle \boldsymbol{z}_{i'_1}, \ldots, \boldsymbol{z}_{i'_D} \rangle\right)}, \tag{1}$$

where $\langle \boldsymbol{z}_{i_1}, \ldots, \boldsymbol{z}_{i_D} \rangle$ is the inner product between D vectors of size K:

$$\langle \boldsymbol{z}_{i_1}, \ldots, \boldsymbol{z}_{i_D} \rangle = \sum_{k=1}^{K} z_{i_1,k} \cdot z_{i_2,k} \cdot \ldots \cdot z_{i_{D-1},k} \cdot z_{i_D,k}. \tag{2}$$

To learn the model parameters, we suppose that a set $\mathcal{X}$ of M observed events $(x_{i_1}, \ldots, x_{i_D})$ is available. This set can also be represented as a D-dimension tensor, in which element $(i_1, \ldots, i_D)$ contains the number of events of $\mathcal{X}$ having context $(i_1, \ldots, i_D)$. We call this structure the *event tensor*.

Considering the events as i.i.d., the observation likelihood of the events in $\mathcal{X}$ corresponds to

$$\begin{aligned} p(\mathcal{X} \mid \mathcal{Z}) &= \prod_{(x_{i_1}, \ldots, x_{i_D}) \in \mathcal{X}} p(x_{i_1}, \ldots, x_{i_D} \mid \mathcal{Z}) \\ &= \prod_{i_1=1}^{N_1} \cdots \prod_{i_D=1}^{N_D} p(x_{i_1}, \ldots, x_{i_D} \mid \mathcal{Z})^{M_{i_1,\ldots,i_D}}, \end{aligned} \tag{3}$$

where $M_{i_1,\ldots,i_D}$ is the number of events of $\mathcal{X}$ with context $(i_1, \ldots, i_D)$. Since the number of events is small compared to the size of the multi-dimensional event space (i.e., $N_1 \times \ldots \times N_j$), only a few of these values are expected to be non-zero. In other words, the event tensor should be very sparse. To regularize the solution, we suppose the factor vectors as independent and following a zero-mean normal distribution with uniform variance:

$$p(\mathcal{Z}) \;=\; \prod_{j=1}^{D} \prod_{i_j=1}^{N_j} \mathcal{N}\big(\boldsymbol{z}_{i_j};\, \mathbf{0},\, \sigma^{-1} I\big). \tag{4}$$

The latent factors $\mathcal{Z}$ are found using the *maximum a posteriori* (MAP) estimate, which corresponds to maximizing the following cost function:

$$\begin{aligned} f(\mathcal{Z}) \;&=\; \log p(\mathcal{X} \,|\, \mathcal{Z}) \;+\; \log p(\mathcal{Z}) \\ &=\; \sum_{i_1=1}^{N_1} \cdots \sum_{i_D=1}^{N_D} M_{i_1,\ldots,i_D} \log p(x_{i_1}, \ldots, x_{i_D} \,|\, \mathcal{Z}) \;-\; \frac{\sigma}{2} \sum_{j=1}^{D} \sum_{i_j=1}^{N_j} ||\boldsymbol{z}_{i_j}||^2. \end{aligned} \tag{5}$$

Note that this is equivalent to minimizing the KL divergence between the model and empirical distribution of events, as done in [4].

Since the cost function of Eq. (5) is both non-linear and non-concave, obtaining globally optimum parameters is an intractable problem. Therefore, we must optimize it using an iterative approach like the gradient ascent method, which has a linear convergence rate. However, because this function is concave with respect to *each* factor vector, we can instead use Nesterov's accelerated gradient method [15] for which the convergence rate is quadratic.

Unlike gradient ascent, Nesterov's method performs two different steps at each iteration. The first step is a simple gradient ascent step of size η from the current solution $\boldsymbol{z}_{i_j}^{(t)}$ to a intermediate solution $\boldsymbol{y}_{i_j}^{(t+1)}$:

$$\boldsymbol{y}_{i_j}^{(t+1)} \;=\; \boldsymbol{z}_{i_j}^{(t)} \;+\; \eta \frac{\partial f}{\partial \boldsymbol{z}_{i_j}} (\mathcal{Z}^{(t)}). \tag{6}$$

Let $R_{i_1,\ldots,i_D}(\mathcal{Z})$ be the difference between the observed number of events in context $(i_1, \ldots, i_D)$ and the expected one according to parameters $\mathcal{Z}$:

$$R_{i_1,\ldots,i_D}(\mathcal{Z}) \;=\; M_{i_1,\ldots,i_D} - M \cdot p(x_{i_1}, \ldots, x_{i_D} \,|\, \mathcal{Z}). \tag{7}$$

The gradient with respect to $\boldsymbol{z}_{i_j}$ is given by

$$\frac{\partial f}{\partial \boldsymbol{z}_{i_j}} (\mathcal{Z}) \;=\; \sum_{i_1=1}^{N_1} \cdots \sum_{i_{j-1}=1}^{N_{j-1}} \sum_{i_{j+1}=1}^{N_{j+1}} \cdots \sum_{i_D=1}^{N_D} R_{i_1,\ldots,i_D}(\mathcal{Z}) \cdot \hat{\boldsymbol{z}}_{i_j} \;-\; \sigma \boldsymbol{z}_{i_j}, \tag{8}$$

where $\hat{\boldsymbol{z}}_{i_j}$ is defined as the Hadamard (i.e., element-wise) product of all factor vectors expect the one of dimension j:

$$\hat{\boldsymbol{z}}_{i_j} \;=\; \big(\boldsymbol{z}_{i_1} \circ \ldots \circ \boldsymbol{z}_{i_{j-1}} \circ \boldsymbol{z}_{i_{j+1}} \circ \cdots \circ \boldsymbol{z}_{i_D}\big), \tag{9}$$

Algorithm 1. Parameter inference using Nesterov's method

Input: The event tensor $\mathcal{X}$ and latent factor size K;
Input: The regularization parameter σ and initial gradient step size η;
Output: The factor matrices $\mathbf{Z}_1, \ldots, \mathbf{Z}_D$;

1 **for** $j = 1, \ldots, D$ **do**
2 Initialize the rows of $\mathbf{Z}_j^{(0)}$ following $\mathcal{N}(\mathbf{0}, \sigma^{-1} I)$;
3 $\mathbf{Y}^{(0)} := \mathbf{Z}_j^{(0)}$;
4 Set $t := 0$ and $a := 0$;
5 **while** $f(\mathcal{Z}^{(t)})$ *not converged* **do**
6 Reconstruct estimated tensor $\hat{\mathcal{X}}$ (unfolded along dim. 1):
  $\hat{\mathbf{X}}_{(1)} := \frac{1}{T} \exp\left(\mathbf{Y}_1^{(t)} \left(\mathbf{Y}_D^{(t)} \odot \ldots \odot \mathbf{Y}_2^{(t)}\right)\right)$,
  where T is such that sum of elements in $\hat{\mathbf{X}}_{(1)}$ is $M = |\mathcal{X}|$;
7 **for** $j = 1, \ldots, D$ **do**
8 StepOK := *false* ;
9 **while** StepOK = *false* **do**
10 $\boldsymbol{G}_j := \left(\mathbf{X}_{(j)} - \hat{\mathbf{X}}_{(j)}\right) \left(\mathbf{Y}_D^{(t)} \odot \ldots \odot \mathbf{Y}_{j+1}^{(t)} \odot \mathbf{Y}_{j-1}^{(t)} \odot \ldots \odot \mathbf{Y}_1^{(t)}\right) - \sigma \mathbf{Y}_j^{(t)}$;
11 $\mathbf{Z}_j^{(t+1)} := \mathbf{Y}_j^{(t)} + \eta \boldsymbol{G}_j$;
12 **if** $f(\mathbf{Z}_j^{(t+1)}) \leq f(\mathbf{Y}_j^t) + \mathrm{Tr}\left(\boldsymbol{G}_j^\top \left(\mathbf{Z}_j^{(t+1)} - \mathbf{Y}_j^t\right)\right) + \frac{1}{2\eta} ||\mathbf{Z}_j^{(t+1)} - \mathbf{Y}_j^t||_F^2$
  then $\eta := 0.5\,\eta$;
13 **else** StepOK := *true* ;
14 $a_{t+1} := \frac{1}{2} + \frac{1}{2}\sqrt{1 + 4\,a_t^2}$;
15 **for** $j = 1, \ldots, D$ **do**
16 $\mathbf{Y}_j^{(t+1)} := \frac{a_{t+1} + a_t - 1}{a_{t+1}} \mathbf{Z}_j^{(t+1)} - \frac{a_t - 1}{a_{t+1}} \mathbf{Z}_j^{(t)}$;
17 $t := t + 1$;
18 **return** $\mathcal{Z} = \{\mathbf{Z}_1^{(t)}, \ldots, \mathbf{Z}_D^{(t)}\}$;

The second step then finds the next solution $\boldsymbol{z}_{i_j}^{(t+1)}$ as a convex combination of the two last intermediate solutions:

$$\boldsymbol{z}_{i_j}^{(t+1)} = (1 - \gamma_t)\, \boldsymbol{y}_{i_j}^{(t+1)} + \gamma_t\, \boldsymbol{y}_{i_j}^{(t)}, \tag{10}$$

where γ_t are constants controlling the search momentum (e.g., see Algorithm 1).

2.2 Algorithm Summary and Complexity

The complete inference process is summarized in Algorithm 1. For a greater efficiency, we group latent factor vectors of each dimension j in a single matrix $\mathbf{Z}_j$, and use standard matrix operations. We start by initializing the factor matrices randomly following the prior distribution of parameter σ (lines 1-3). Then, at each iteration of Nesterov's method, the tensor $\hat{\mathcal{X}}$ of expected counts is reconstructed using the current intermediate factors $\mathbf{Y}_j$ (line 6). Operator $\odot$ corresponds to the

Khatri-Rao product and $\mathbf{X}_{(j)}$ denotes the unfolding of tensor $\mathcal{X}$ along dimension j (see [12] for more information). For each dimension j, the gradient is then computed using the residual between the observed and expected counts (line 10), and used to update factor matrix $\mathbf{Z}_j$ (line 11). If the step size η is too large, the solution may diverge. A strategy is thus added to detect such problem and adjust η automatically (lines 12-13), thereby eliminating the need to tune η manually. This strategy is known as *backtracking line search*. The momentum constant and intermediate factors are finally updated as per Nesterov's method (lines 14-16). The process is repeated until converge is attained, or a maximum number of iterations is exhausted.

The computational complexity of this algorithm is as follows. For each iteration, reconstructing the expected tensor $\hat{\mathcal{X}}$ take $O(K \cdot \prod_{j=1}^{D} N_j)$ operations. Likewise, updating the latent factors for each dimension can be done in $O(K \cdot \prod_{j=1}^{D} N_j)$. Therefore, the total complexity is $O(T_{\max} \cdot K \cdot D \cdot \prod_{j=1}^{D} N_j)$, where $T_{\max}$ is the maximum number of iterations.

2.3 Abnormal Event Detection

We use the latent factors learned during training to evaluate the joint probability of new events in real-time, and mark as abnormal those that have a low probability. Let θ be a given probability threshold, an event $(x_{i_1}, \ldots, x_{i_D})$ will be marked as abnormal if $p(x_{i_1}, \ldots, x_{i_D} \mid \mathcal{Z}) < \theta$. By pre-computing the denominator of Eq. (1), evaluating the joint probability of an incoming event requires only $O(D \cdot K)$ operations.

To detect specific types of anomalies, we can instead evaluate the probability of a single dimensional value, conditioned on all other dimensional values. For example, we could evaluate the probability that the event occurs in a certain zone, given the person, day, and time of day corresponding to that event. If the conditional probability of the observed zone is much lower than that of other zones, the event would then be marked as abnormal. Suppose, without loss of generality that the query dimension is $j = 1$. The probability of x_{i_1}, conditioned on all other dimensions, is given by

$$p(x_{i_1} \mid x_{i_2}, \ldots, x_{i_D}, \mathcal{Z}) \;=\; \frac{\exp\left(\langle \boldsymbol{z}_{i_1}, \ldots, \boldsymbol{z}_{i_D} \rangle\right)}{\sum_{i_1=1}^{N_1} \exp\left(\langle \boldsymbol{z}_{i_1}, \ldots, \boldsymbol{z}_{i_D} \rangle\right)}. \tag{11}$$

To evaluate the computational complexity of this query, we note that the inner product can be decomposed as $\langle \boldsymbol{z}_{i_1}, \ldots, \boldsymbol{z}_{i_D} \rangle = \langle \boldsymbol{z}_{i_1}, \hat{\boldsymbol{z}}_{i_1} \rangle$. Thus, if we pre-compute $\hat{\boldsymbol{z}}_{i_1}$, each inner product computation has a time complexity in $O(K)$, where K is the size of the latent subspace. Since we have to compute N_1 of these inner products, the total cost of evaluating the query is in $O(N_1 \cdot K)$.

3 Experiments

We evaluated the performance of our method by conducting three sets of experiments, related to the low-rank approximation of tensors, the prediction of future events, and the detection of contextual anomalies.

3.1 Low-Rank Approximation

The goal of this first experiment is to measure the ability of our method to fit the event tensor using a small number of parameters.

Experimental Design. We generated two synthetic datasets, each containing 11 sparse 3D tensors of size 100×100×100 drawn from two types of distributions: log-linear and Poisson. Each tensor was generated using three sets of 100 latent factor vectors of size $K = 20$, drawn from a Gaussian prior in the case of log-linear tensors and a Gamma prior for Poisson tensors. The parameters of these priors were selected to have a sparsity level between 70% and 90%, and the resulting tensors normalized to have a total number of events equal to $M = 10^6$.

We used the first tensor to tune the regularization parameter σ of our method, and the remaining 10 to evaluate its average performance in terms of Mean Absolute Error (MAE). Factor sizes of $K = 5, 10, 15, 20$ were tested. We compared our Log-Linear Tensor Factorization (LLTF) method to two state-of-the-art factorization approaches: Alternating Poisson Regression (APR) [4] and Non-Negative Tensor Factorization (NTF) [21]. The Matlab Tensor Toolbox v2.5 [2] implementation of these methods was used.

Results and Discussion. Figure 1 (left) gives the average MAE obtained by the three tested approaches on the Poisson (blue curves) and log-linear (green curves) tensors. As expected, the reconstruction error decreases with higher values of K. Moreover, our LLTF method outperforms APR and NTF for log-linear tensors, especially for $K = 20$ where LLTF obtains an average MAE 2.26 times smaller than APR and 2.41 smaller than NTF. For Poisson data, LLTF performs as well as APR, even though this data is tailored to APR's Poisson model not LLTF's log-linear one. Since it is not designed for sparse count data, NTF obtains a lower performance than LLTF and APR for both log-linear and Poisson data.

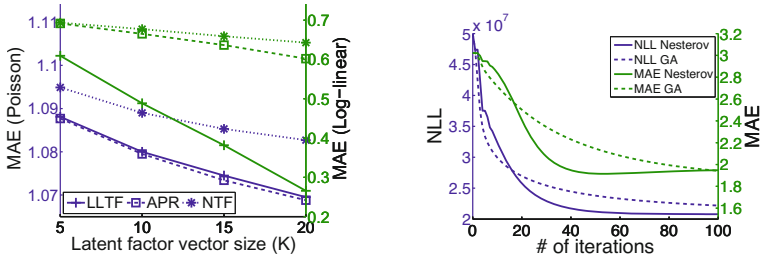

Fig. 1. (**left**) Average MAE obtained by LLTF, APR and NTF on 10 Poisson (blue curves) and log-linear (green curves) tensors, for latent factor sizes of $K = 5, 10, 15, 20$. (**right**) Convergence of our Nesterov-based method on a sample tensor, compared to simple gradient ascent (GA).

Figure 1 (right) illustrates the convergence rate of our Nesterov-based method on a sample tensor, compared to simple gradient ascent (GA). We see that the convergence in terms of Negative Log-Likelihood (NLL) (blue curves) is attained within 60 iterations, with an average time of 0.3 seconds per iteration, whereas GA has not converged after 100 iterations. In contrast to our method, APR takes on average 1.3 seconds per iteration using the same hardware, and requires over 1000 iterations to converge.

3.2 Future Event Prediction

The second experiment evaluates how well our method can predict the number of future events occurring in a given context. As the event tensor is sparse, this experiment measures the ability of the model to predict events that were not observed in the training data.

Experimental Design. Two real-life datasets were used for this experiment.

- **Reality Mining** [6]: Contains the tracking information of 106 students and faculty members from the MIT Media Laboratory and Sloan Business school, collected through their cell-phones in 2004-2005. From the 106 participants, we picked 87 students as the remaining ones had either less than 7 days of data or no data at all. For the locations, we used the 1027 unique cell-tower IDs, corresponding to the attribute *areaID.cellID* in the data. The timestamps of the tracking events were encoded using 24 discrete values, one for each hour of the day. Combining these three dimensions, we obtained a $87 \times 1027 \times 24$ tensor, each cell containing the number of times a person was recorded as being near a given cell tower, at a given time of the day.
- **Geo-Life Taxi Trajectories** [22]: Contains the GPS trajectories of 10,357 taxis during the period of Feb. 2 to Feb. 8, 2008 within the city of Beijing. From the 10,357 taxis in the data, we selected 259 taxis as the remaining ones had either less than 5000 records of temporal locations or no records at all. Since most records are located near the center of Beijing, we converted the Cartesian coordinates (longitude and latitude) to log polar ones (log radius, angle θ) using the city's center as origin. We divided θ into 12 bins of 30° each, and the log radius into 10 bins, giving a total of 120 zones. Similar to the Reality Mining dataset, we encoded the timestamps using 24 discrete values, one for each hour of the day. Combining these three dimensions, we obtained a $259 \times 120 \times 24$ tensor, each cell containing the number of times a taxi was recorded as being in a specific zone, at a given hour of the day.

We split the datasets temporally, putting the first 60% of each person or taxi's events in the training set, the following 20% in the validation set, which was used to tune the regularization parameter σ, and the remaining 20% in the test set. We predicted the number of events in the test set for each context (i.e., tensor cell) by multiplying the probability obtained for this context during training with the total number of events in the test set. We evaluated the prediction accuracy of our method, in terms of MAE, RMSE (Root Mean Squared Error) and NLL, and compared it once again to with APR and NTF. Note that the Poisson distribution used in APR is specifically tailored to model count data. A latent factor size of $K = 10$ was used for all three methods.

Results and Discussion. The prediction accuracy of the three tested methods on the Reality Mining and Taxi datasets is detailed in Table 1. We see that LLTF outperforms APR and NTF, on both datasets and all three performance metrics. Thus, LLTF obtains a MAE 2.32 times lower than APR in the Reality Mining dataset, and 2.10 times lower in the Taxi dataset. We also note that APR obtained infinite NLL values. This is because it gave a zero probability to the events in the test set that were not observed in the training set. By regularizing the factors, our model can better predict such unobserved events.

Table 1. Prediction error obtained by our LLTF method, as well as the NTF and APR approaches, on the Reality Mining (with $\sigma = 6$) and Taxi (with $\sigma = 4$) datasets. NLL values have been scaled for convenience.

(a) **Reality Mining**

Metric	LLTF	APR [4]	NTF [21]
MAE	0.384	0.892	0.894
RMSE	8.906	19.134	19.743
NLL $\times 10^6$	7.270	∞	7.814

(b) **GeoLife Taxi**

Metric	LLTF	APR [4]	NTF [21]
MAE	1.691	3.549	3.084
RMSE	15.319	19.657	24.220
NLL $\times 10^6$	9.158	∞	9.719

3.3 Abnormal Event Detection

In this last experiment, we assess the usefulness of our method to detect abnormal events from their context.

Experimental Design. Once again, the Reality Mining and GeoLife Taxi Trajectory datasets were considered for this experiment. Three types of synthetic anomalies were generated.

- **Swap People:** This type of anomalies simulates a person (or taxi) behaving like someone different. To generate such anomalies, we first computed the KL divergence between the event distribution (i.e., number of events for each time and zone) of all pairs of persons. We then used weighted sampling to pick random person pairs, those with a higher KL divergence having a greater chance of being selected, and swapped the *personID* of all events involving the corresponding two persons.
- **Swap Times:** This type of anomalies corresponds to a person going to the same places, but at odd times (hours of the day). To generate these anomalies, we randomly picked a person and computed the KL divergence between the event distribution (number of events for each zone) of all time pairs, for this person. Once more, the time pairs were picked using the KL divergence as sampling weight, and the *timeID* of these pairs were swapped in all the events of the selected person.
- **Swap Zones:** This last type anomaly corresponds to a person being active at the same times, but in unusual zones. These anomalies were generated using the same strategy as in *Swap Times*, except that the KL divergence between the time distribution of events of each zone pair was considered.

We split the dataset as in the previous experiment, and used the training data as is to learn the distribution of normal events. For both the validation and testing sets, we generated 10 different sets of random anomalies, using the following procedure. For *Swap People* anomalies, we swapped 5 pairs of people/taxis, while for *Swap Times* and *Swap Zones* anomalies, we randomly picked 3 persons and, for each of them, swapped 3 pairs of *timeID* or *zoneID*. The parameters of the tested methods were tuned using the average Area Under the ROC Curve (AUC) obtained over the 10 validation anomaly sets. Note that this tuning step was necessary to have a fair comparison between the tested methods, but such validation anomalies may not be available in real-life applications. Finally, the performance of the tuned methods was evaluated as the mean AUC obtained over the 10 test anomaly sets.

We tested four variations of our proposed approach. In the first one, called LLTF-Joint, the ROC curves are generated by evaluating the joint probability of test examples, as defined in Eq. (1), and then computing the precision/recall for increasing probability thresholds. The other three methods, denoted by LLTF-D, where $D = \{\text{Person}, \text{Time}, \text{Zone}\}$, instead use the conditional probability of a single dimension D given the other two dimensions, as described in Eq. (11).

We compared the performance of these methods with two well-known unsupervised anomaly detection approaches: One-Class Support Vector Machines (OCSVM) [16] and Kernel Density Estimation (KDE) [13]. For both of these methods, the discrete dimensional values (e.g., *personID*) were first converted to binary features using an indicator function, giving a total of 1138 binary features for Reality Mining and 403 binary features for Taxi. PCA was then applied to these binary features, using a percentage of variance value of 95%, and the resulting components were normalized to have uniform variance. For OCSVM, two parameters required tuning: ν, which controls the fraction of training examples allowed outside the learned region, and the RBF kernel parameter γ. The signed distance to the hyperplane was used to evaluate the normality of test examples, while computing the ROC curves. For KDE, we evaluated the probability of a test example $\boldsymbol{x}$ (projected in PCA space) as

$$p(\boldsymbol{x}) \propto \frac{1}{N}\sum_{n=1}^{N} \exp\left\{-\frac{1}{h}||\boldsymbol{x}-\boldsymbol{x}_n||^2\right\}, \tag{12}$$

where $\boldsymbol{x}_n$ are the training examples (in PCA space) and h is the kernel bandwidth parameter, tuned on the validation data.

Results and Discussion. The mean ROC curves (and corresponding AUC values), computed over the 10 test sets of each anomaly type, are shown in Figure 2. Except for the *Swap Time* anomalies, our LLTF-Joint method obtained a higher mean AUC than OCSVM and KDE. In particular, the AUC of LLTF-Joint is 9% to 35% higher than OCSVM, and 10% to 37% higher than KDE, for *Swap People* anomalies. Furthermore, the conditional probability model can improve the detection of specific types of anomalies. For instance, LLTF-Person obtained mean AUC values of 0.97 and 0.93 on the Swap People anomalies, compared to 0.95 and 0.92 for LLTF-Joint. Similarly, LLTF-Time obtained a mean AUC of 0.85 on the Swap Time anomalies of the Reality Mining dataset, whereas this value was only 0.63 for LLTF-Joint.

The proposed method is also faster and more robust than OCSVM and KDE. Thus, training LLTF for the Taxi dataset took less than 10 minutes on a Quad-Core AMD 2.3 GHz processor with 8 GB of RAM, whereas training OCSVM on this dataset required over 7 hours using the same hardware (no training is necessary for KDE). Likewise, predicting anomalies in the test set took on average 0.03 *ms* for LLTF, compared to 61 *ms* for OCSVM and 7 *ms* for KDE. Moreover, while the best parameters for each type of anomaly (as selected in validation) varied greatly in OCSVM and KDE, our method was more robust to the choice of parameters: $K = 25$, $\sigma = 8$ was used for *all* anomalies in the RM dataset, and $K = 15$, $\sigma = 6$ for *all* anomalies in Taxi.

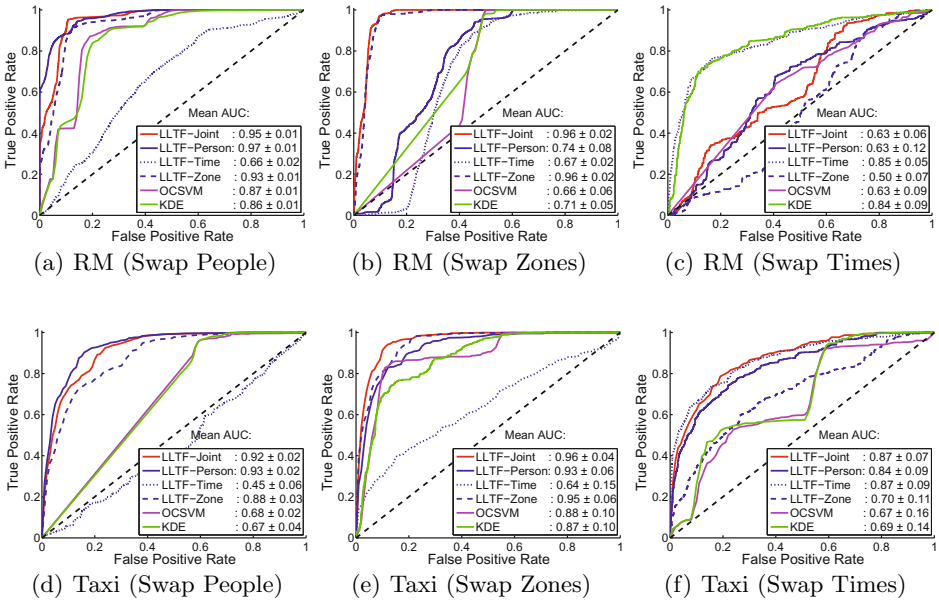

Fig. 2. Average ROC curves and AUC obtained by our LLTF, as well as the OCVSM and KDE approaches, on 10 sets of synthetic anomalies generated from the Reality Mining (RM) and Taxi datasets. Three types of anomalies are considered: swapping the events of two people/taxis, and swapping the time/zone of events corresponding to a person/taxi.

4 Conclusion

We presented a new approach, based on log-linear tensor factorization, for the detection of contextual anomalies. A parametric model was proposed to estimate the joint probability of dimensional values, in which the parameters are the factors of an event count tensor. To learn the factors, an efficient technique based on Nesterov's accelerated gradient ascent was presented. The proposed approach was evaluated on three problems: the low-rank approximation of synthetic tensors, the prediction of future of events in real-life data and the detection of events representing abnormal behaviors. Results show our method to outperform state of the art approaches for these problems, while being faster and more robust than these approaches. As future work, we will investigate the use of additional dimensions in the tensor, for instance to model the duration and sequence of events, and extend the method to perform online learning.

References

1. Bader, B.W., Berry, M.W., Browne, M.: Discussion tracking in enron email using parafac. In: Survey of Text Mining II, pp. 147–163. Springer (2008)
2. Bader, B.W., Kolda, T.G., et al.: Matlab tensor toolbox version 2.5 (2012). http://www.sandia.gov/~tgkolda/TensorToolbox/

3. Benezeth, Y., Jodoin, P.M., Saligrama, V., Rosenberger, C.: Abnormal events detection based on spatio-temporal co-occurences. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 2458–2465 (2009)
4. Chi, E.C., Kolda, T.G.: On tensors, sparsity, and nonnegative factorizations. SIAM Journal on Matrix Analysis and Applications **33**(4), 1272–1299 (2012)
5. Chung, P.C., Liu, C.D.: A daily behavior enabled hidden markov model for human behavior understanding. Pattern Recognition **41**(5), 1572–1580 (2008)
6. Eagle, N., Pentland, A.: Reality mining: sensing complex social systems. Personal and Ubiquitous Computing **10**(4), 255–268 (2006)
7. Hara, K., Omori, T., Ueno, R.: Detection of unusual human behavior in intelligent house. In: 12th IEEE Workshop on Neural Networks for Signal Processing, pp. 697–706 (2002)
8. Hayashi, K., Takenouchi, T., Shibata, T., Kamiya, Y., Kato, D., Kunieda, K., Yamada, K., Ikeda, K.: Exponential family tensor factorization for missing-values prediction and anomaly detection. In: 10th IEEE Int. Conf. on Data Mining (ICDM), pp. 216–225 (2010)
9. Hu, D.H., Zhang, X.X., Yin, J., Zheng, V.W., Yang, Q.: Abnormal activity recognition based on hdp-hmm models. In: IJCAI, pp. 1715–1720 (2009)
10. Jiang, F., Yuan, J., Tsaftaris, S.A., Katsaggelos, A.K.: Anomalous video event detection using spatiotemporal context. Computer Vision and Image Understanding **115**(3), 323–333 (2011)
11. Kim, H., Lee, S., Ma, X., Wang, C.: Higher-order PCA for anomaly detection in large-scale networks. In: 3rd IEEE Int. Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), pp. 85–88 (2009)
12. Kolda, T., Bader, B.: Tensor decompositions and applications. SIAM Review **51**(3), 455–500 (2009)
13. Latecki, L.J., Lazarevic, A., Pokrajac, D.: Outlier detection with kernel density functions. In: Perner, P. (ed.) MLDM 2007. LNCS (LNAI), vol. 4571, pp. 61–75. Springer, Heidelberg (2007)
14. Li, C., Han, Z., Ye, Q., Jiao, J.: Abnormal behavior detection via sparse reconstruction analysis of trajectory. In: 6th IEEE Int. Conf. on Image and Graphics (ICIG), pp. 807–810 (2011)
15. Nesterov, Y.: Gradient methods for minimizing composite functions. Mathematical Programming **140**(1), 125–161 (2013)
16. Piciarelli, C., Micheloni, C., Foresti, G.L.: Trajectory-based anomalous event detection. Circuits and Systems for Video Technology **18**(11), 1544–1554 (2008)
17. Rendle, S., Gantner, Z., Freudenthaler, C., Schmidt-Thieme, L.: Fast context-aware recommendations with factorization machines. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 635–644. ACM (2011)
18. Sillito, R.R., Fisher, R.B.: Semi-supervised learning for anomalous trajectory detection. In: BMVC, pp. 1–10 (2008)
19. Sun, J., Tao, D., Faloutsos, C.: Beyond streams and graphs: dynamic tensor analysis. In: 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 374–383 (2006)
20. Tork, H.F., Oliveira, M., Gama, J., Malinowski, S., Morla, R.: Event and anomaly detection using tucker3 decomposition. In: Workshop on Ubiquitous Data Mining, p. 8 (2012)
21. Welling, M., Weber, M.: Positive tensor factorization. Pattern Recognition Letters **22**(12), 1255–1261 (2001)
22. Yuan, J., Zheng, Y., Xie, X., Sun, G.: Driving with knowledge from the physical world. In: 17th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 316–324 (2011)