Robust Multivariate Autoregression for Anomaly Detection in Dynamic Product Ratings

Nikou Günnemann Stephan Günnemann Christos Faloutsos

Carnegie Mellon University, USA

{nguennem, sguennem, christos}@cs.cmu.edu

ABSTRACT

User provided rating data about products and services is one key feature of websites such as Amazon, TripAdvisor, or Yelp. Since these ratings are rather static but might change over time, a temporal analysis of rating distributions provides deeper insights into the evolution of a products' quality.

Given a time-series of rating distributions, in this work, we answer the following questions: (1) How to detect the base behavior of users regarding a product's evaluation over time? (2) How to detect points in time where the rating distribution differs from this base behavior, e.g., due to attacks or spontaneous changes in the product's quality? To achieve these goals, we model the base behavior of users regarding a product as a *latent* multivariate autoregressive process. This latent behavior is mixed with a sparse anomaly signal finally leading to the observed data. We propose an efficient algorithm solving our objective and we present interesting findings on various real world datasets.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications —*Data mining*; I.2.6 [**Artificial Intelligence**]: Learning

Keywords

anomaly detection; robust autoregression; sparsity

1. INTRODUCTION

Rating scores about products and services are ubiquitous in today's websites such as Amazon, Yelp, or TripAdvisor. These ratings supply new customers information about the products' quality and support the decision making process which product to buy or service to use. In today's era of fast-paced developments in industry and growing competitions between companies and manufacturers, these ratings can also influence a companies production line. In fact, the ratings of customers can be considered as a benchmark for future sales performance of a product. To avoid negative results, companies can derive benefits from customer ratings by detecting functional weaknesses as well as deficiencies of products for improving the detected lacks.

In this work, we propose a method for the temporal analysis of rating distributions. Given a time-series of rating distributions, our goal is to extract the *base behavior* of users regarding the product's quality over time as well as to discover time points at which the product's evaluation shows *anomalous patterns*. The base behavior represents the general quality of a product accounting for temporal effects like, e.g., decreasing quality due to technical progress of competing products. The anomalies, in contrast, represent irregularities where the observed ratings deviate from the base behavior. These anomalies might occur, for example, due to spammers trying to push the success of a product or due to changes in the product's manufacturing process.

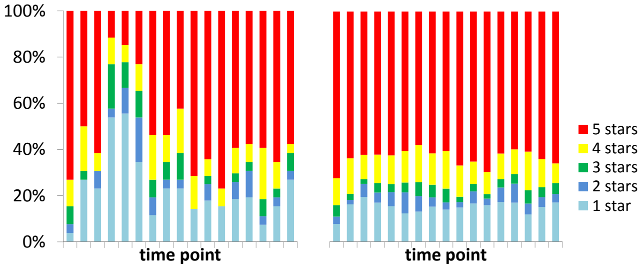

Figure 1: Left: Observed rating distributions of the product "coconut-water" over time. Right: Detected base behavior without anomalies.

A real world example for such an effect is illustrated in Figure 1 (left). The diagram illustrates the distribution of ratings over time for a "coconut water" sold on Amazon (more details about the data are given in the experimental section). The different colors represent the fraction of the star ratings – from 1 star (light blue) to 5 stars (red) – at a certain time. At the very beginning and at the majority of the later time points, the product received good ratings. During the time period 4-6, however, the rating distributions behave differently: one observes a high increase of negative ratings. As we will show in Section 5, these anomalies occurred due to a change in the products packaging – the previously used paper bottles were replaced by plastic bottles leading to an unpleasant aftertaste.

With our method, we aim at detecting such anomalies as well as the corresponding base behavior if the data would be "anomaly free". Let us anticipate and present the result of our method: As shown in Figure 1 (right), our method successfully detects the base behavior of users regarding the

WWW'14, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2744-2/14/04.
http://dx.doi.org/10.1145/2566486.2568008.

product, and, as we will see, it will provide further information about the anomalies and the points in time when they have been occurred.

Besides using our method to detect weaknesses in products and services, it can generally be used to find points in time where irregular ratings have been given. Thus, by filtering out these information new customers might be provided with a more precise evaluation of the product or – as another extreme – these irregular ratings can specifically be used to provide the whole picture on a product (since these anomalous ratings are otherwise hidden in the much larger set of normal behavior). Furthermore, our method allows to predict the rating distributions of future time steps exploiting the underlying base behavior. Thus, when new ratings arrive one can assess whether they match the predicted values, i.e. normal behavior, or whether they deviate, thus, indicating an anomaly.

The general idea of our method is to consider the base behavior as a *latent* multivariate autoregressive process, thus, incorporating the temporal dynamics of the data. The base behavior is then mixed with a latent, sparse anomaly signal to finally generate the observed data.

The contributions of our work are:

- *Novel mining task:* We present a technique for the temporal analysis of product ratings that detects the base behavior of users as well as potential anomalies enriched with their points in time when they occurred. Furthermore, our technique can be used to predict the rating distributions at future points in time.
- *Theoretical soundness:* We base our method on a sound generative process that models dynamic rating distributions by mixing a latent autoregressive process with a latent and sparse anomaly signal.
- *Algorithm design:* We develop an efficient algorithm that solves our objective by invoking a sequence of quadratic programs. As a further benefit, our method does not require user-defined parameters.
- *Effectiveness:* We evaluate our method on different real world datasets and we show its effectiveness by presenting interesting findings.

While the rating of products or services has been studied extensively in other research areas such as, e.g., recommender systems and opinion mining (cf. Section 4), the goals of these areas are completely different to our work. To the best of our knowledge, there are no works that consider the temporal analysis of rating distributions incorporating potentially anomalous behavior.

2. THE RLA MODEL

In this section, we introduce our RLA model (Robust Latent Autoregression) for detecting the users' base rating behavior and anomalies in rating distributions. Following convention, we do not distinguish between a random variable X and its realization $X = x$ if it is clear from the context. Vectors of (random) variables are written in bold font, e.g. $\boldsymbol{b}$, while the entries of the vectors are given in standard font, e.g. b_i. The number of time steps is denoted with T and the number of dimensions with D.

Before formally introducing our objective, we discuss the data, used as the input for our model, in more detail.

Preliminaries - Data Representation. In our work, we aim at analyzing the temporal evolution of rating distributions. We assume that users can choose ratings based on an ordinal rating scale with M different ratings (e.g. star ratings from 1 to M). Correspondingly, a distribution over these ratings can be represented by a M-dimensional vector $\mathbf{r} \in [0 \dots 1]^M$ with $\sum_{d=1}^{M} r_d = 1$, where r_i denotes the fraction of ratings with value i. The raw data we process is a time-series $\mathbf{R} = (\mathbf{r}^{(1)}, \dots, \mathbf{r}^{(T)})$ of such rating distributions, i.e. a multivariate time-series of length T.

Since ratings are given on an ordinal scale, we use the established principle of analyzing the *cumulative distributions* instead of the raw data [2]. That is, instead of considering $\mathbf{r}^{(t)}$, we analyze the cumulative distributions $\mathbf{x}^{(t)}$ with $x_d^{(t)} = \sum_{d'=1}^{d} r_{d'}^{(t)}$. By analyzing the cumulative distributions we preserve the ordering of the ratings and we can better describe differences in rating distributions.

Consider, e.g., the (non-cumulative) distributions $\mathbf{a} = [1, 0, 0, 0]$, $\mathbf{b} = [0.5, 0.5, 0, 0]$, and $\mathbf{c} = [0.5, 0, 0, 0.5]$. Intuitively, the distributions $\mathbf{a}$ and $\mathbf{b}$ are more similar to each other than $\mathbf{a}$ and $\mathbf{c}$, since $\mathbf{a}$ represents only 1-star ratings, $\mathbf{b}$ 1&2-star ratings and $\mathbf{c}$ 1&4-star ratings. By considering the cumulative distributions $\mathbf{a}' = [1, 1, 1, 1]$, $\mathbf{b}' = [0.5, 1, 1, 1]$, and $\mathbf{c}' = [0.5, 0.5, 0.5, 1]$ this similarity structure is directly visible (even without doing cross-dimension comparisons).

Since the last entry of the cumulative distribution is always 1, we can safely ignore it from our considerations. Additionally, to be a valid cumulative distribution, the values of $x_d^{(t)}$ have to be non-decreasing in d. As an abbreviation for later use, we define

$$\mathcal{C}_D := \{\mathbf{x} \in [0 \dots 1]^D \mid \forall i : x_i \leq x_{i+1}\}$$

Overall, in the remainder of this paper, we consider the data $\mathbf{X} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)})$ where each $\mathbf{x}^{(t)}$ has dimensionality $D := M - 1$ and $\mathbf{x}^{(t)} \in \mathcal{C}_D$.

For easier interpretation, when plotting data, we always use the non-cumulative distribution.

Generative Process. Given the observed time series $\mathbf{X} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)})$, our aim is to extract the base behavior of the users and the corresponding points in time where anomalies occur. The challenge of this task is that the observed data $\mathbf{X}$ is already polluted by anomalies, thus, directly using it for estimating the base behavior might be misleading.

In our model, we solve this issue by assuming that the observed time series $\mathbf{X} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)})$ is obtained by mixing the base (but unknown) user behavior $\mathbf{A} = (\mathbf{a}^{(1)}, \dots, \mathbf{a}^{(T)})$ with an (also unknown) anomaly behavior $\mathbf{y}$. Thus, the base behavior $\mathbf{A}$ acts as a *latent* variable which is not directly observed but inferred by our technique.

An overview of our generative process showing the used variables and their dependencies is illustrated by the graphical model in Figure 2. More formally, we assume that the observed data follows the random generative process

$$\mathbf{x}^{(t)} = p_t \cdot \mathbf{a}^{(t)} + (1 - p_t) \cdot \mathbf{y} + \boldsymbol{\varepsilon}_t \quad \forall t = 1 \dots T \qquad (1)$$

where $p_t \in [0 \dots 1]$ is the mixing coefficient at time t and $\boldsymbol{\varepsilon}_t$ corresponds to white noise (e.g. a normal distribution). The higher p_t the stronger the effect of the base user behavior at a certain point. Thus, the vector $\mathbf{p}$ is effectively the indicator where the anomalies have been occurred. To ensure the interpretability of the model, we require that the base behavior as well as the anomaly behavior are valid cumulative distributions, i.e. $\mathbf{a}^{(t)}, \mathbf{y} \in \mathcal{C}_D$.

At this point it is important to highlight the difference between outliers and anomalies – two different kinds of irregular data (since this difference is not consistently handled in the literature, we describe our notion here). *Outliers* are irregular behavior that can be attributed to mostly random corruptions of the data (like, e.g., measurement errors). *Anomalies*, in contrast, are irregular behavior that follow a specific pattern (like, e.g., time points with consistently low ratings due to a change in the product's quality). In our method, we consider anomalous behavior, whose characteristic is described by the distribution $\mathbf{y}$.

While the observed data $\mathbf{X}$ might show abrupt changes due to the anomalies, we assume that the base behavior changes smoothly over time. An established principle to model such behavior is the use of (vector) autoregressive models [20, 21], where the values observed at time t depend on the values of previously observed time points allowing slight perturbations. Based on this general idea, we define the base behavior to be generated by the random process

$$\mathbf{a}^{(t)} = w \cdot \mathbf{a}^{(t-1)} + (1-w) \cdot \mathbf{b} + \hat{\boldsymbol{\varepsilon}}_t \quad \forall t = 1 \ldots T \tag{2}$$

Here, $w \in [0 \ldots 1]$ determines the importance of the previous time step, $\mathbf{b} \in \mathcal{C}_D$ is a vector, which represents the trend, and $\hat{\boldsymbol{\varepsilon}}_t$ is again white noise.[1] The value $\mathbf{a}^{(0)}$, required for a valid recursion, represents the base behavior of the users at (the virtual) time point 0.

We want to mention two important differences of Equation 2 in contrast to usual autoregressive models: First, we use a convex combination of the vectors $\mathbf{a}^{(t-1)}$ and $\mathbf{b}$, i.e. we use weights w and $w' = 1 - w$. In general, autoregressive models allow unbounded values for w and w'. The convex combination leads to the huge advantage of generating only "valid" data, i.e. if $\mathbf{a}^{(t-1)}$ and $\mathbf{b}$ represent valid (cumulative) distributions, the convex combination is also a valid (cumulative) distribution. This property is in particular useful, when using the autoregressive model for predicting future time steps: in this case, the error term will be equal to zero and, thus, the convex combination leads to a prediction representing a valid distribution. As we will see in the experimental analysis, this aspect is neglected by all competing approaches making their results difficult to interpret.

Second, unlike to traditional autoregressive models, where the values of $\mathbf{a}$ are observed (and thus fixed), we consider a *latent* autoregressive model. Thus, similar to the values of w, and $\mathbf{b}$, the values of $\mathbf{a}$ are variables.

Even though Equation 1 and 2 look similar at a first sight, their underlying principle is very different. While the weights p_t might change over time to allow different strength of anomalies, the weights w are constant reflecting the smooth base behavior. While the value of $\mathbf{x}^{(t)}$ depends only on values of the same time point, the value of $\mathbf{a}^{(t)}$ depends on the values of the previous time point. And most important, while the values of $\mathbf{x}^{(t)}$ are observed, the values of $\mathbf{a}^{(t)}$ are latent.

Sparsity. So far, we assumed that the values of p_t are independent of each other. Thus, scenarios where we might observe "anomalies" at every point in time ($p_t < 1$ for all t) are possible. Since such a scenario would contradict the idea of an anomaly, it is reasonable to require that anomalies are rare events. Technically speaking, we want to enforce the vector $\mathbf{1} - \mathbf{p}$ to be sparse.

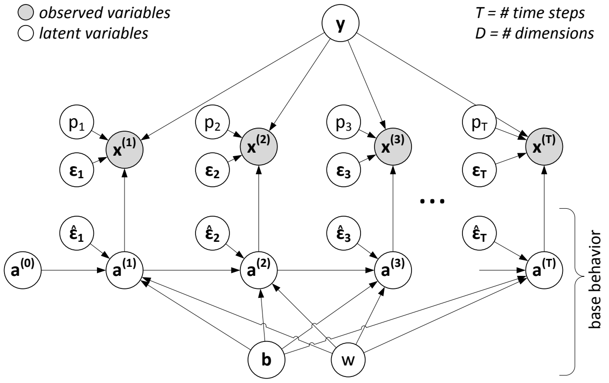

Figure 2: Graphical model of RLA. The observed data is a mixture of the (potentially evolving) base behavior $\mathbf{a}^{(t)}$ with the anomaly behavior $\mathbf{y}$.

Let λ denote the maximal number of anomalies a user expects in the data. We require $\|\mathbf{1} - \mathbf{p}\|_0 \leq \lambda$ where $\|.\|_0$ denotes the L$_0$ pseudo-norm and $\mathbf{1}$ is the vector consisting of only ones. In this case, the variables p_t are no longer independent since we effectively use the prior distribution

$$p(\mathbf{p}) \propto \begin{cases} const & \text{if } \|\mathbf{1} - \mathbf{p}\|_0 \leq \lambda \\ 0 & \text{else} \end{cases} \tag{3}$$

from which the values of $\mathbf{p}$ are drawn. For illustration purposes we do not show this dependency in our graphical model in Figure 2.

The above prior distribution is beneficial due to multiple reasons: First, it is easy to interpret. By using the L$_0$ norm, the value of λ represents simply an upper bound on the number of anomalies and one can precisely spot the time steps where an anomaly has occurred. Second, the distribution is essentially the maximum-entropy distribution taking into account the upper bound of λ anomalies. That is, it is a non-information prior which is not biased to certain $\mathbf{p}$ vectors. Finally, as we will show later, by using the above prior we can use a model selection approach to determine the parameter λ automatically. Thus, the user does not need to provide it by hand.

Objective. The ultimate goal of our method is to infer the values of the hidden variables which best describe the observed data. There are multiple ways to formulate this objective. In this work, we are interested in finding the maximum likelihood solution w.r.t. the complete data. That is, we aim at maximizing the joint probability $p(\mathbf{X}, Z)$ where Z denotes the set of all variables. To make this problem well-defined we finally have to select appropriate prior distributions. We simply assume independent, identically distributed Gaussian error, i.e. $\boldsymbol{\varepsilon}_t, \hat{\boldsymbol{\varepsilon}}_t \sim \mathcal{N}(0, \sigma^2 \cdot I)$. In this case, the variables $\boldsymbol{\varepsilon}_t, \hat{\boldsymbol{\varepsilon}}_t$ can be absorbed into Equation 1 and Equation 2 by writing

$$\mathbf{x}^{(t)} \sim \hat{\mathcal{N}}(p_t \cdot \mathbf{a}^{(t)} + (1-p_t) \cdot \mathbf{y}, \sigma^2 \cdot I)$$

$$\mathbf{a}^{(t)} \sim \hat{\mathcal{N}}(w \cdot \mathbf{a}^{(t-1)} + (1-w) \cdot \mathbf{b}, \sigma^2 \cdot I)$$

Here, $\hat{\mathcal{N}}$ denotes the Normal distribution restricted to the domain $\mathcal{C}_D$ since we require $\mathbf{a}^{(t)}, \mathbf{x}^{(t)} \in \mathcal{C}_D$.

[1] Equation 2 is based on a first-order autoregressive process. Extensions to higher order processes are straightforward, though, not focus of this work.

Additionally, we select non-informative priors for the values of $\mathbf{y}$, $\mathbf{a}^{(0)}$, $\mathbf{b}$, and w. That is,

$$p(\mathbf{a}^{(0)}) \propto \begin{cases} const & \text{if } \mathbf{a}^{(0)} \in \mathcal{C}_D \\ 0 & \text{else} \end{cases}$$

and accordingly for $\mathbf{y}$, $\mathbf{b}$, and w. Note that these are valid priors since the domain of the vectors is bounded.

Using this setting, we can derive that the likelihood, if all variables are in their valid domains, is proportional to

$$\begin{aligned} p(\mathbf{X}, Z) &\propto \prod_{t=1}^{T} p(\mathbf{a}^{(t)}|\mathbf{a}^{(t-1)}, \mathbf{b}, w) \cdot p(\mathbf{x}^{(t)}|\mathbf{a}^{(t)}, \mathbf{y}, \mathbf{p}) \qquad (4) \\ &\propto \frac{1}{\sigma^{D \cdot T \cdot 2}} \cdot exp(-\frac{1}{2 \cdot \sigma^2} f(\mathbf{a}^{(0)}, \ldots, \mathbf{a}^{(T)}, \mathbf{b}, \mathbf{y}, w, \mathbf{p})) \end{aligned}$$

where the function f is given as

$$\begin{aligned} f(...) := \sum_{t=1}^{T} \Big(&\left\| \mathbf{x}^{(t)} - p_t \cdot \mathbf{a}^{(t)} - (1 - p_t) \cdot \mathbf{y} \right\|_2^2 \\ &+ \left\| \mathbf{a}^{(t)} - w \cdot \mathbf{a}^{(t-1)} - (1 - w) \cdot \mathbf{b} \right\|_2^2 \Big) \end{aligned}$$

Solving for the optimal value of σ^2 that maximizes this equation leads to $\sigma^2 = \frac{f(\ldots)}{2 \cdot D \cdot T}$ (assuming $f \neq 0$, otherwise $\sigma^2 \to 0$ is the optimal solution). Using this result in Equation 4, the likelihood finally becomes proportional to $1/f(\mathbf{a}^{(0)}, \ldots, \mathbf{a}^{(T)}, \mathbf{b}, \mathbf{y}, w, \mathbf{p})^{D \cdot T}$. Thus, overall, maximizing Equation 4 corresponds to solve the following objective

$$\min_{\mathbf{a}^{(0)}, \ldots, \mathbf{a}^{(T)}, \mathbf{b}, \mathbf{y}, w, \mathbf{p}} f(\mathbf{a}^{(0)}, \ldots, \mathbf{a}^{(T)}, \mathbf{b}, \mathbf{y}, w, \mathbf{p}) \qquad \text{(P1)}$$

subject to

$$\|\mathbf{1} - \mathbf{p}\|_0 \leq \lambda \ \wedge \ \mathbf{a}^{(t)}, \mathbf{b}, \mathbf{y} \in \mathcal{C}_D \ \wedge \ 0 \leq w \leq 1 \ \wedge \ 0 \leq p_t \leq 1$$

This formulation intuitively states that the optimal solution is the one that minimizes the (squared) residuals regarding the observed data and the base user behavior.

Model Selection. Solving problem P1 leads to the optimal solution w.r.t. a specific upper bound λ on the number of anomalies. To determine the value of λ, we use a model selection principle. Since we use the L_0 pseudo-norm as our constraint, λ effectively controls the number of free parameters in our model. Increasing λ by a value of one, increases the number of free parameters by two: We allow one additional p_t value to vary, plus we have the freedom to decide at which time t this happens.[2] One exception represents the step from $\lambda = 0$ to $\lambda = 1$, which increases the number of free parameters by $2 + D$. The additional term D accounts for the vector $\mathbf{y}$ which is not used in the case of $\lambda = 0$.

Given these observations, we use the Bayesian information criterion [5] to determine the optimal value of λ. That is, we choose the λ minimizing

$$BIC(\lambda) = -2 \cdot \ln L_\lambda + k_\lambda \cdot \ln(D \cdot T)$$

where $k_\lambda = m + 2 \cdot \lambda + D \cdot \min(\lambda, 1)$ is the number of free parameters determined based on the observations made above. Here, the term m represents the remaining parameters of our model which are not effected by the choice of λ and which, therefore, do not effect the choice of the optimal λ (i.e. for the model selection step we can simply set $m = 0$). The term L_λ is the likelihood of the data which in our case simplifies to $L_\lambda = 1/f(\mathbf{a}^{(0)}, \ldots, \mathbf{a}^{(T)}, \mathbf{b}, \mathbf{y}, w, \mathbf{p})^{D \cdot T}$ (cf. above).

Special Cases. We briefly want to discuss two special cases of our model. First, when fixing $p_t = 1$ for all t, our model reduces to a special case of Kalman filtering [5], which is not able to detect the anomaly signal $\mathbf{y}$. Thus, if anomalies are present in the data, the estimates of the Kalman Filter might be highly corrupted. Second, when fixing $w = 0$, we ignore the temporal effects of the data modeled via autoregression. In this case, only the vectors $\mathbf{b}$ and $\mathbf{y}$ can be used to generate the data. When additionally dropping the sparsity constraint, the data corresponds to a mixture of two static components as similar done by classical matrix factorization techniques. Further relations to existing methods can be found in Section 4.

Prediction. Besides detecting the base behavior and anomalies for the given data, our method can also be used to predict the rating distributions of future points in time. Assuming that anomalies are rare, it is natural to exploit only the base behavior for the prediction step. Thus, the most likely (cumulative) distribution at time $T + 1$ is

$$\tilde{\mathbf{x}}^{(T+1)} = w \cdot \mathbf{a}^{(T)} + (1 - w) \cdot \mathbf{b} \qquad (5)$$

Here, since a-priori no information about the error at the next time step is known, the original error terms have been replaced with their expected value of 0. Comparing the observed distribution at time $T + 1$ against the predicted one gives an indicator whether a new anomaly has been observed.

3. ALGORITHM

Finding an exact solution to problem P1 is infeasible due to two reasons: First, albeit natural, the sparsity constraint via the L_0 norm is NP-hard to optimize in general [8, 24]. Second, the objective function is neither convex nor concave, which prevents the (direct) application of efficient convex optimization solvers. Thus, in the following, we present an algorithm computing a near-optimal solution. An overview of our method is given in Algorithm 1.

Sparsity. One standard solution to tackle the complexity of the L_0 norm is to replace it with the L_1 norm. As shown in [7], however, an iterative reweighting of the L_1 norm outperforms the simple (unweighted) L_1-norm in many situations. In preliminary experiments, this effect has also been observed in our scenario: by using the unweighted L_1-norm some values in the $\mathbf{1} - \mathbf{p}$ vector were close to (but not exactly) 0 leading to a bad approximation of the L_0-norm.

Thus, we follow the study of [7] and we replace the L_0 norm with the weighted L_1 norm, where the weights are iteratively updated based on the previously determined solution. This principle leads to a sequence of optimization problems (cf. line 12 of Algorithm 1). Technically, we use the constraint

$$\|Z \cdot (\mathbf{1} - \mathbf{p})\|_1 \leq \tau \qquad (6)$$

as a proxy for $\|\mathbf{1} - \mathbf{p}\|_0 \leq \lambda$, where $Z = diag(z_1 \ldots, z_T)$.

Initially, all weights z_t are set to 1, corresponding to the unweighted L_1 norm. After solving our objective using this constraint, the weights are recomputed based on $\mathbf{p}$:

$$z_t = \frac{1}{1 - p_t + \delta} \cdot \max_t(1 - p_t + \delta) \qquad (7)$$

[2] It is fair to mention that the value 2 is a slight overestimate: Due to the inequality constraint $\leq \lambda$, we essentially sample the timepoints where anomalies occur with replacement. Thus, with some probability we sample an already drawn point. Being aware of this effect, we can conclude that our model selection approach might slightly underestimate the optimal value of λ.

where δ is a very small constant to ensure that the fraction is always well defined (default $\delta = 10^{-4}$). The idea is that values p_t which are close to 1 will get higher weights z_t. Therefore, in the next iteration of our method, they are stronger penalized when deviating from 1 (effectively steering these values to become exactly 1). An analytical justification for this principle is given in [7].

It is worth mentioning that Equation 7 differs from the equation proposed in [7] by introducing the multiplicative factor $\max_t(1-p_t+\delta)$. Since in [7] the weighted norm is used as the *objective function*, any (positive) multiplicative factor would lead to the same result. In our scenario, however, we use the weighted L_1 norm as a *constraint*. If we would allow arbitrary small values for z_t, the constraint effectively becomes useless, i.e. it is no longer guaranteed that $\mathbf{1}-\mathbf{p}$ is sparse. In contrast, by using our definition it holds that $z_t \geq 1$ and, therefore,

$$\|Z \cdot (\mathbf{1}-\mathbf{p})\|_1 \geq \|(\mathbf{1}-\mathbf{p})\|_1 \tag{8}$$

Indeed, the factor $\max_t(1-p_t+\delta)$ leads to the tightest possible solution that fulfills the above equation. Combining Equation 8 with Equation 6, it becomes clear that during each iteration of our method we guarantee

$$\|(\mathbf{1}-\mathbf{p})\|_1 \leq \|Z \cdot (\mathbf{1}-\mathbf{p})\|_1 \leq \tau$$

and, therefore, the sparsity of the vector $\mathbf{p}$ is realized.

Since the values of p_t (and thus $1-p_t$) are bounded between 0 and 1, Equation 6 can directly be written as

$$\sum_{t=1}^{T} z_t \cdot (1-p_t) \leq \tau \quad \Leftrightarrow \quad \sum_{t=1}^{T} -z_t \cdot p_t \leq \tau - \sum_{t=1}^{T} z_t \tag{9}$$

which corresponds to a simple, linear constraint on the values of $\mathbf{p}$.

Reduction to quadratic programs. While the objective function of our original problem definition is not jointly convex in all variables, we can observe the following: Given the values of $\mathbf{p}$ and w, the objective function is convex in $\mathbf{a}^{(*)}$, $\mathbf{y}$ and $\mathbf{b}$; and simultaneously, given $\mathbf{a}^{(*)}$, $\mathbf{y}$, and $\mathbf{b}$, the objective function is convex in $\mathbf{p}$ and w.

We show an even stronger result: both problems (i.e. conditioned either on $\mathbf{p}, w$ or $\mathbf{a}^{(*)}, \mathbf{y}, \mathbf{b}$) are instances of a quadratic program.

THEOREM 1. *Given $\mathbf{p}$ and w. Let $(\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_T, \hat{\mathbf{b}}, \hat{\mathbf{y}}, \mathbf{p}, w)$ be the optimal solution of problem P1 (when fixing the values of $\mathbf{p}$ and w). Set $\hat{\mathbf{u}} = stack(\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_T, \hat{\mathbf{b}}, \hat{\mathbf{y}})$, where stack denotes the stacking of multiple vectors to a single one.*

There exist vectors $\mathbf{c}^\bullet$, $\mathbf{d}^\bullet$ and (sparse) matrices $\mathbf{Q}^\bullet$, $\mathbf{A}^\bullet$ such that

$$\hat{\mathbf{u}} = \arg\min_{\mathbf{u}\in\mathbb{R}^{(T+3)\cdot D}} \frac{1}{2}\cdot\mathbf{u}^T\cdot\mathbf{Q}^\bullet\cdot\mathbf{u}+(\mathbf{c}^\bullet)^T\cdot\mathbf{u} \text{ subject to } \mathbf{A}^\bullet\cdot\mathbf{u} \leq \mathbf{d}^\bullet$$

PROOF. See appendix □

THEOREM 2. *Given $\mathbf{a}^{(*)}, \mathbf{y}$, and $\mathbf{b}$. Let $(\mathbf{a}^{(0)}, \ldots, \mathbf{a}^{(T)}, \mathbf{b}, \mathbf{y}, \hat{\mathbf{p}}, \hat{w})$ be the optimal solution of problem P1 (when fixing the values of $\mathbf{a}^{(*)}, \mathbf{y}, \mathbf{b}$). Set $\hat{\mathbf{v}} = stack(\hat{\mathbf{p}}, \hat{w})$.*

There exist vectors $\mathbf{c}^\circ$, $\mathbf{d}^\circ$ and (sparse) matrices $\mathbf{Q}^\circ$, $\mathbf{A}^\circ$ such that

$$\hat{\mathbf{v}} = \arg\min_{\mathbf{v}\in\mathbb{R}^{T+1}} \frac{1}{2}\cdot\mathbf{v}^T\cdot\mathbf{Q}^\circ\cdot\mathbf{v}+(\mathbf{c}^\circ)^T\cdot\mathbf{v} \text{ subject to } \mathbf{A}^\circ\cdot\mathbf{v} \leq \mathbf{d}^\circ$$

PROOF. See appendix □

```
   Data: series of rating distributions $\mathbf{X} = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(T)})$
1  $z_1 = \ldots = z_T = 1$;
2  initialize $\mathbf{p}, w$;
3  until convergence do
4      until convergence do
5          recompute $\mathbf{Q}^\bullet, \mathbf{c}^\bullet$ ;
6          determine $\hat{\mathbf{u}}$ via quad. programming (cf. Theo. 1);
7          set $\mathbf{a}^{(*)}, \mathbf{b}, \mathbf{y}$ based on $\hat{\mathbf{u}}$;
8          recompute $\mathbf{Q}^\circ, \mathbf{c}^\circ$ ;
9          determine $\hat{\mathbf{v}}$ via quad. programming (cf. Theo. 2);
10         set $\mathbf{p}, w$ based on $\hat{\mathbf{v}}$ ;
11     end
12     update weights $z_t$ based on Eq. 7
13 end
```

Algorithm 1: The RLA algorithm

The above two results make it possible to use highly efficient solvers for quadratic programming in our scenario, in particular by exploiting the fact that the matrices are sparse.

Overall method. As shown in Algorithm 1, our method uses a block coordinate-descent to alternatively optimize the variables. In each step it invokes a quadratic programming solver (line 6 & 9). After updating one set of variables, we recompute the corresponding matrices/vectors according to Theorem 1/2, which subsequently are used to compute the optimal solution for the other set of variables. Per default, we initialize $\mathbf{p}$ with high values since we expect only few anomalies in the data. w is initialized randomly. We assume convergence if the relative change in the objective function value is less than 0.01%. Overall, our method allows to efficiently compute a near-optimal solution of problem P1.

4. RELATED WORK

The main objective of our work is to spot anomalies in dynamic rating data by simultaneously detecting the users' base behavior evolving through the time. In the following, we discuss methods related to our approach. We first discuss methods that are related to our method from a technical perspective (i.e. the methodology of modeling the data is related). Afterwards, we give a broader perspective on methods whose application domain is related to ours.

Related Techniques. *Autoregression* (AR) models [21] represent data as a random process based on previous time points as well as noise. An AR model can be written as $x_t = \sum_{m=1}^{M} w_m \cdot x_{t-m} + \epsilon$ where x is the point series under investigation, M is the order of the AR model, w_m are some coefficients taking arbitrary values and finally ϵ is noise. While traditional autoregression models only consider univariate data, multivariate extensions (so called *vector autoregression* (VA) models) have been proposed [20, 18].

In contrast to our method, where the autoregression model is used as a latent process, the existing techniques consider directly the observed data. Thus, these approaches are not robust to anomalies in the data. It is a well known fact that in these models an anomaly (or in general a so-called shock) effects all time points infinitely far into the future. Thus, these models fail to find good approximations of the data corrupted by anomalies.

Kalman Filter/Smoother [5] can be considered as a generalization of first-order vector autoregressive models. While Kalman Filters are very flexible in generating the data, they share a similar drawback as AR/VA models. They are sensitive to anomalies in the data. The relation to our model to Kalman Filter has already been discussed in Section 2.

Robust extensions of vector autoregression [9] and Kalman Filtering [25] to handle outliers have been proposed. These methods try to circumvent errors in the data by, e.g., downweighting the effect of high residuals (which usually occur due to outliers) via, e.g., the Huber loss function.

Again we want to highlight the difference between outliers and anomalies as already described in Section 2. *Outliers* are abnormal behavior attributed to mostly independent, random corruptions of the data, while *anomalies* are abnormal behavior following a specific pattern. It is clear that outliers and anomalies are two different concepts which should be handled differently. While all the above mentioned methods handle outliers, our method is specifically designed to handle anomalies.

Finally, as already described in Section 2, our model uses convex combinations of the involved variables, thus ensuring that the final results are valid distributions and leading to easy interpretability. The above techniques do not enforce such a constraint leading to potentially questionable results.

We compare our method against Kalman Filtering, a vector autoregression technique [18] and an extension to handle outliers [9] in our experimental section.

Matrix decomposition techniques (e.g., PCA) are designed to decompose a data matrix into products or sums of other matrices exhibiting special structure (e.g. low-rank or sparsity). These techniques do not consider the temporal characteristics of the data but they treat each time point as an independent observation, thus, ignoring highly important information. In particular, these techniques can not be used to predict future points in time based on the current observations. Similar as the temporal methods described above, the standard versions of these methods are not robust to anomalies or outliers. While *robust extensions* considering outliers have been proposed [6, 26], these methods are still not aware of the data's temporal characteristics.

In the experimental section, we compare our RLA method against ICA (Independent Component Analysis) [10], PCA (Principal Component Analysis) [13], NNMF (Non-Negative Matrix Factorization) [4], and robust NNMF [26], the latter one being a robust extension to handle outliers.

Related Applications. Our method can be used to detect anomalies in a time-series of rating distributions and to describe the base behavior of users regarding the considered product. In the following we discuss related applications.

Multiple techniques, considering various data types, have been proposed in the area of *outlier detection* [1]. While the majority of techniques tackles the case of independently distributed data, time-series outlier detection and outlier detection for streaming data are also an active field of research [1]. Both areas differ from our work. In time-series outlier detection one considers a set of time-series, where some of the time-series might be outliers. In contrast, we operate on a single multivariate time-series where some points in time might be anomalous. Streaming outlier detection assumes that the data under considering is not completely given but successively arrives over time – additionally coping with further aspects as limited storage of the data. In our work, however, we assume the complete history of data is given, thus, allowing enhanced analysis capabilities. Again, we have to note that most existing techniques consider outlier in the sense of independent, random errors in the data. We, in contrast, assume the anomalies to be generated from a specific anomalous behavior.

Change detection techniques try to detect points in time where the state of the underlying system has changed [16, 14]. A change might not generally indicate anomalous behavior. Indeed, since we model the base behavior as an autoregressive model, even the base behavior might change over time. Thus, change detection techniques cannot directly solve the problem tackled in this work.

The study of product ratings has been done in multiple research areas, all following different goals and objectives. *Recommender Systems* often incorporate ratings and their temporal information [12, 15, 22] to improve the prediction performance. Their goal is to recommend products or services to users which they most likely are going to use. *Opinion mining* or sentiment analysis aims at extracting the sentiment of users regarding specific products or features of a product [19, 3, 23]. This way, opinion mining can be considered as a technique to infer the ratings. *Modeling* of social rating networks, e.g. to compactly describe the underlying mechanism driving the network or to generate synthetic data, has been studied in, e.g., [11, 17].

None of the existing methods is designed to detect anomalies in a time-series of rating distributions under consideration of a potentially evolving base behavior.

5. EXPERIMENTAL ANALYSIS

In the following, we empirically analyze our RLA method.

Datasets. We applied our system on over hundred thousands of product ratings representing a variety of categories: 1) We used an extract of the Amazon website evaluating different food products[3] (400k ratings). 2) We analyzed ratings of restaurants/services in the area of Phoenix based on an extract of the Yelp website[4] (230k ratings). 3) Using an extract of the TripAdvisor website[5] (250k ratings), we used our method for studying hotel ratings.

The data consists of the IDs of the products/services to be rated as well as the related user IDs who evaluated them with star rating scores from 1 up to 5 at different timesteps (in the case of TripAdvisor, the rating scores range from 0 up to 5). Additionally, these datasets contain textual reviews, which we used to understand and describe the results of our method. Since our method is designed to handle rating distributions, we can flexibly process the raw data by, e.g., using equi-width binning or equi-depth binning. We decided to use equi-depth binning (each 25 temporally successive ratings are aggregated) to avoid sparsity effects.

Besides these real world datasets we used synthetic data generated based on the presented process to analyze the scalability and robustness of our method.

Runtime analysis. We start with a brief runtime analysis. The runtime of our method is primarily affected by two characteristics: the length T as well as the dimensionality D (i.e. the number of different ratings) of the time-series. To analyze these effects, we generated time-series of different length (100 to 10,000) with a small percentage of anomalies (5%) according to our model. We used the dimensionalities 4, 5, and 6 since these values reflect the properties of the frequently used rating schemes. All experiments were conducted on commodity hardware with 3 GHz CPU's and 4 GB main memory.

[3] http://snap.stanford.edu/data/

[4] http://www.yelp.com/dataset_challenge/

[5] http://sifaka.cs.uiuc.edu/~wang296/Data/

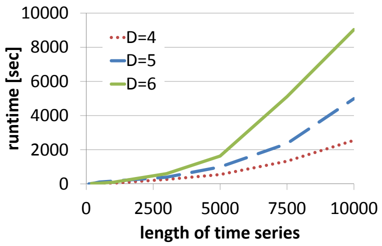

Figure 3: Runtime vs. length of time-series

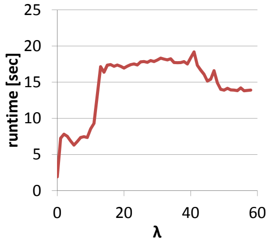

Figure 4: Runtime vs. λ (sparsity)

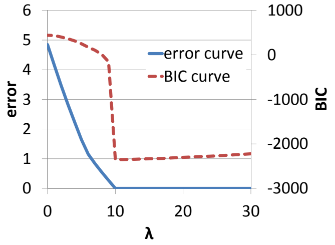

Figure 5: Error & BIC vs. λ (synthetic data)

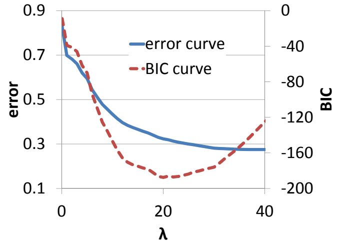

Figure 6: Error & BIC vs. λ (real world data)

The results are illustrated in Figure 3. The runtime increases approximately quadratic in the length of the time-series; similar to (or even better than) many matrix factorization techniques. While the quadratic complexity might be theoretically a problem for very long time-series, we want to mention that the absolute runtimes are small. For a length of 10,000 (which would already correspond to more than 27 years of data when measured on a daily basis – longer than most rating websites exist), the absolute runtime for, e.g., 5 ratings is around 80 minutes on commodity hardware.

In Figure 4, we analyze the effect of a varying λ on the runtime. In the figure, we plotted the results for an exemplary hotel from the TripAdvisor database. We observe an interesting behavior: Starting from very small values of λ, an increase in λ also increases the runtime. This effect might be attributed to the larger amount of variables the model now needs to fit. Given a certain value of λ, however, the runtime behaves almost constantly. Even more interestingly, when selecting a very high value, the runtime starts decreasing again. This last effect might be explained as follows: when choosing very large λ values, the p_t values can be optimized almost independently. Decreasing the value of λ, however, increases the dependency, thus, leading to higher runtimes. We observed a similar behavior across the various datasets on all of the three studied databases.

Effectiveness and Robustness. Next, we analyze the effectiveness and robustness of our method. We again start with synthetic data (100 time points, 10 anomalies, 5 dimensions). Figure 5 shows the minimal sum-of-squared-errors value (i.e. the value of the function f; cf. P1) obtained by our method when varying the parameter λ. Obviously, when increasing the value of λ, the error decreases since more flexibility is provided. We observe a high decrease in the error until reaching the value of $\lambda = 10$, corresponding to the true number of anomalies in the data. Afterward, the gain of allowing further anomalies decreases. The absolute error obtained by our method is small, showing that we well describe the underlying data.

On the second y-axis of Figure 5, we illustrate the BIC values. As shown, the minimal BIC value is obtained for the value of 10. Thus, the BIC gives a good guideline which λ parameter to choose.

Figure 6 shows the same type of plot for a real world dataset (hotel Punta Cana Princess, TripAdvisor – the original data is shown in Figure 7, left). Again, the absolute error converges to a very small value; our method leads to a good approximation. Additionally, the BIC curve clearly indicates which λ to choose. In Figure 7, the middle diagram shows the detected base behavior for this product. The behavior evolves smoothly over time and shows the general trend for this hotel containing of primarily high ratings. Mixing this base behavior with the detected anomalies leads to the right plot in Figure 7, which almost perfectly recovers the original data.

In Figure 8 we analyze for the same dataset how the error decreases with an increasing number of iterations of our algorithm until convergence (i.e. on the x-axis we count how often the inner-loop of Algorithm 1 has been executed, while the y-axis shows the obtained error). We parametrized our method with different λ values to show different effects. As expected, for all λ values, the first iterations (around 10) lead to the highest decrease of the error. Interesting to note are the sudden increases in the error for some of the curves at some points in time. At these points in time, the inner-loop of Algorithm 1 has been converged, thus, leading to an update of the weights enforcing the sparsity (Eq. 7). After updating the weights, our method is forced to select other (more sparse) solutions for $\mathbf{p}$, which obviously increase the error. After each sudden increase, the error term again decreases corresponding to the iterations of the inner-loop of our algorithm. Indeed, these increases show that the outer loop of our method is actually effective, i.e. it steers the $\mathbf{p}$ vector to other solutions. Furthermore, the figure shows

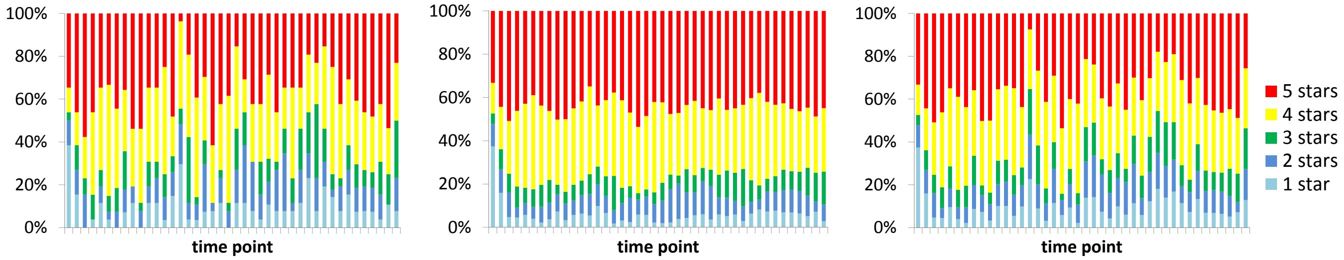

Figure 7: Left: Rating distributions over time for the hotel Punta Cana Princess (TripAdvisor data). Middle: Detected base behavior. Right: Reconstructed data after mixing with the anomaly signal.

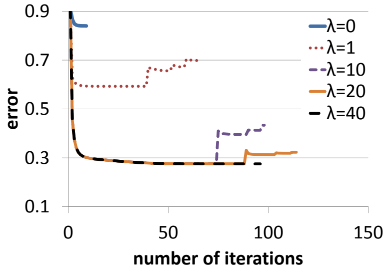

Figure 8: Convergence analysis

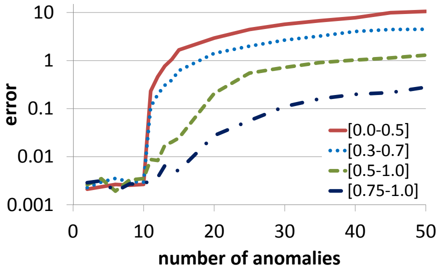

Figure 9: Robustness (here: $\lambda = 10$)

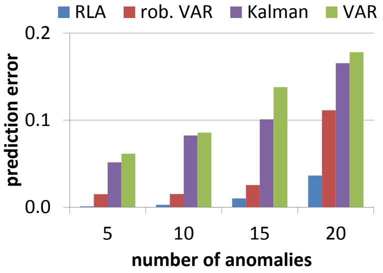

Figure 10: Prediction error

that only few iterations of the outer-loop are necessary to reach convergence. All variants converge in less than 120 iterations. In particular, the variant allowing no anomalies (λ=0) converges very quickly; though, of course, with a high error. As already shown in Fig. 6, increasing the value of λ lowers the obtained error since more flexibility is obtained.

In the next experiment, we analyze the robustness of our technique. For this experiment, we generated synthetic data with a varying number of anomalies (from 2-50). Additionally, we varied the strength of the anomalies by allowing the vector $\mathbf{p}$ to take only values from a specific domain: from very strong anomalies (p values between 0-0.5) to weak anomalies (p values from 0.75 to 1). For this experiment, we fixed the value of λ to 10, to show how our algorithm copes with data which is corrupted more than expected.

Figure 9 shows the result of this experiment. The x-axis shows the number of anomalies in the data, while the y-axis shows the reconstruction error of our method. As expected, if the number of anomalies in the data is less than 10, our method almost perfectly recovers the data. When exceeding this threshold, the error successively increases. The stronger the anomalies, the stronger the increase in the error. If the anomalies are weak, the errors can somehow be compensated by the base behavior.

Comparison with related techniques. In the following, we compare our RLA method against competing approaches. Due to the heterogeneity of the methods, we first describe the setup. As mentioned in Sec. 4, we compare against the (non-temporal) approaches PCA, ICA, and NNMF. To make these techniques applicable in our scenario – distinguishing between normal and irregular behavior –, we parametrize these methods to detect two components. For the temporal vector-autoregression approaches, we use an order of one, as our model does, too. The transition model and noise model of the Kalman Filter are learned via the EM algorithm.

We compare the methods based on their reconstruction error (i.e. how well these techniques approximate the data). For PCA, ICA, Kalman Filtering, and the non-robust variants of NNMF and vector autoregression, computing the approximated data is straightforward. Similarly, it is easy to compute the approximation using the result of our technique by simple mixing the base behavior with the anomaly signal. The robust NNMF and robust VAR, however, do not allow such an easy approach: Informally, these approaches remove points from the data when they are regarded as outliers, and the outliers do not need to follow a specific pattern (in contrast to our anomalies). Thus, given the results of these techniques one cannot simply reconstruct the original data. The principle we follow here is to replace the vectors (or entries) that have been regarded as outliers by these techniques with the original data. Thus, assuming for these points optimal reconstruction. Obviously, this gives a huge advantage to the competing methods.

Given the approximated data $\hat{\mathbf{X}}$, we compute the values $err = \|\hat{\mathbf{X}} - \mathbf{X}\|_F^2/T$, i.e. the approximation error per time stamp. This relative error allows us to compute the average over multiple time-series with varying length.

One advantage of our approach is the focus on convex combinations of valid distributions, thus, leading to only valid distributions for the approximated data (cf. Section 2 and 4). None of the competing methods considers this aspects. In many cases, we observed that the approximated vectors did not sum up to 1, thus, representing invalid distributions. An interpretation of such results is difficult or even misleading. Thus, in a second step, we additionally normalize the results of the competing methods to become valid distributions, leading to the normalized approximated data $\hat{\mathbf{X}}_{norm}$. Accordingly, we also compute the error $err_{norm} = \|\hat{\mathbf{X}}_{norm} - \mathbf{X}\|_F^2/T$.

		synth. data		real world data	
		err	err_{norm}	err	err_{norm}
temporal	RLA	0.0006	0.0006	0.0177	0.0177
	VAR	0.0098	0.0098	0.0368	0.0368
	rob. VAR	0.0060	0.0060	0.0330	0.0330
	Kalman	0.0091	0.0091	0.0273	0.0273
non-temp.	NNMF	0.0040	0.0044	0.0239	0.0252
	rob. NNMF	0.0027	0.0030	0.0213	0.0223
	PCA	0.0040	0.0044	0.0239	0.0252
	ICA	0.0102	0.0102	0.0242	0.0242

Table 1: Reconstruction error of the competing methods on synthetic and real world data.

Table 1 shows the results of all methods. In the left two columns ones sees the results on a synthetic dataset with 10 anomalies and 100 time points, the right two columns show the error averaged over multiple real world datasets. Considering the synthetic data one sees that RLA clearly outperforms the other temporal algorithms. In particular, the (non-robust) VAR and Kalman Filtering are sensitive to the anomalies in the data. Considering the non-temporal methods, ICA is not able to recover the hidden structure in the data. The error of the robust NNMF method is the smallest among the non-temporal approaches; though, as explained above, this method has an advantage when computing the error. Still, the error of this technique is larger than the one of RLA.

For real world data, we can infer similar results: Our RLA technique obtains the smallest error and, in particular, outperforms the other temporal techniques.

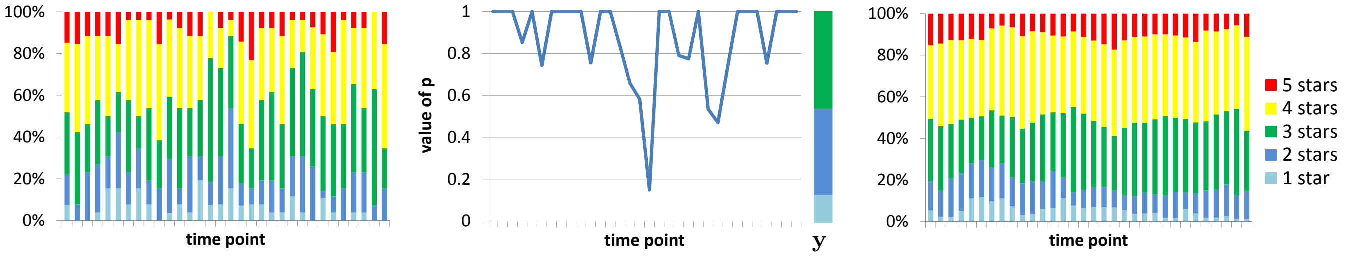

Figure 11: Left: Rating distributions over time for the Phoenix Airport (Yelp data). Middle: p vector and detected anomaly signal y. Right: Detected base behavior.

Comparison via prediction. The temporal techniques allow to predict the rating distribution at future points in time. In the following experiment, we analyze the methods' prediction accuracy. We generated synthetic data with 100 time points and an increasing number of anomalies, where we ensured that the rating distribution $\mathbf{x}$ at the last point in time is not an anomaly. We then removed $\mathbf{x}$ from the data, we applied the techniques on the remaining data, and we predicted the future rating distribution (e.g. in our model according to Equation 5). Similar as above, we compute the error $err_{norm} = \|\hat{\mathbf{x}}_{norm} - \mathbf{x}\|_F$, where $\hat{\mathbf{x}}_{norm}$ is the (normalized) predicted distribution.

Figure 10 shows the results. The non-robust VAR and Kalman Filtering show the highest error. In particular, for a high number of anomalies their error increases quickly since the hidden structure of the data can no longer be detected. RLA is more robust to the anomalies and also outperforms the robust VAR method. Obviously, for all methods a higher number of anomalies is more challenging when predicting the future rating distribution.

Discoveries. So far, we surveyed the efficiency and accuracy of our algorithm. In the following, we will demonstrate the application of our algorithm by illustrating some of our interesting discoveries from the three investigated real world datasets. Our approach for extracting products with noticeable rating irregularities relies on strong fluctuations in the $\mathbf{p}$ vector. As described in Section 2, small values in $\mathbf{p}$ weight the anomaly behavior strongly, thus, reflecting irregularities in the rating behavior. In contrast, the higher p_t the stronger the effect of the base user behavior at a certain point.

(1) As described in the introduction, one of our discoveries from the Amazon-Food data is the product "coconut-water"[6], presented in Figure 1. Figure 12 shows the values of the product's $\mathbf{p}$ vector at different timesteps t (x-axis). It is clear that during the timesteps $4-6$, the elements of $\mathbf{p}$ take small values which reflect the irregularities in the rating behavior. Additionally, Figure 12 shows the detected anomaly distribution $\mathbf{y}$. While the base behavior (Figure 1 (right)) shows the general trend of the product with primarily good ratings, the anomaly clearly explains the high amount of negative ratings observed during the time $4-6$. In particular, the anomaly shows a zero percentage of 5 star ratings. A closer look at the product's reviews during this time explains this behavior: Most of the customers complain about the *"new plastic bottle"* and *"a bad after taste"*. Since the manufacturer has recently changed his production from paper bottles to plastic bottles, many customers were disappointed. However, reviews after the anomalous time points hint to an improvement in the production (e.g., *"I can understand a lot of the initial bad reviews as I thought the new plastic bottle had a bad after taste. ... I can say that the taste is much improved ..."*).

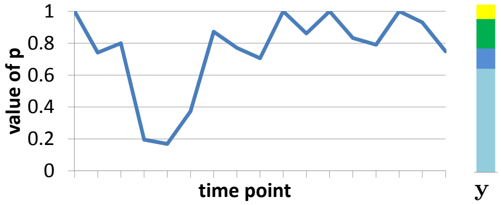

Figure 12: p vector and anomaly behavior y for the product "coconut-water" (Amazon data).

(2) Another interesting discovery of our algorithm from the Amazon-Food dataset is the detection of high anomalies in the rating of a dog food product[7]. The studied reviews at the detected time points show that the anomalies occurred due to the change of the country of manufacturing from the USA to another country: *"Originally, these were made in US. Now made in xxx ... advertised as natural, cage free, no hormones, no fillers, no antibiotics ...HUH? Not in xxx... I will not trust these as safe treats"*.

(3) Analyzing the dataset Yelp, an interesting pattern is found for the Sky Harbor Airport in Phoenix. Figure 11 (middle) shows the $\mathbf{p}$ vector for the rating behavior. Clearly, the elements of the $\mathbf{p}$ vector reach small values at the two intervals $[15, 18]$ and $[23, 25]$. Comparing this plot with the original rating data in Figure 11 (left), we are able to spot the sub-populations which were identified as irregularities by our method. Ignoring these irregularities, the base rating behavior looks as presented in Figure 11 (right). It is clear, that the rating behaves relatively robust with mostly 3-4 star ratings. The anomalous behavior, in contrast, shows a much higher fraction of 2 stars, and almost zero percentage of 4-5 stars. Mixing the base behavior with the anomalous signal using the $\mathbf{p}$ vector, our algorithm supplies an approximation to the real data with a very small residual error of $5.75 \cdot 10^{-4}$ per time stamp.

Analyzing the different reviews at yelp, it is noticeable that the main reasons for the high anomalies are, e.g., *"not many choices for food"* and *"long walkway between terminals"*. However, reviews shortly after the second anomaly note that *"... they are now doing some construction to try and fix things ..."*, giving hint to an improvement of the airport which may have caused the better evaluation (almost no 1 star ratings) at the end of the rating behavior.

(4) In Figure 7 we have already illustrated the distribution of ratings over time for the hotel Punta Cana Princess

[6] http://www.amazon.com/dp/B000CNB4LE

[7] http://www.amazon.com/dp/B001410U50

evaluated on TripAdvisor. While manually detecting irregularities for this data might be difficult, examining the distribution of the p_t values (cf. Figure 13), clearly shows that there are 5 time intervals at which most of the anomalies are positioned. For these points the generally very good rating of the hotel (cf. Figure 7 (middle)), heavily drops. This behavior is reflected in the anomaly signal $\mathbf{y}$. Comparing the noticeable p_t values with the corresponding reviews, it is recognizable that the number of negative criticizes such as *"lack of communication"* and *"poor service and bad quality of food"* highly increases. Using such information in dependency of their time of occurrence can be used by the hotel's management to improve the quality of their service.

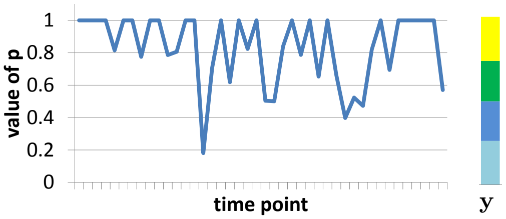

Figure 13: p vector and anomaly behavior y for hotel Punta Cana Princess (TripAdvisor).

Discoveries via prediction. Since our method exploits the temporal properties of the data, it allows us to predict future points in time. To show this potential of our algorithm for the scenario of anomaly detection, we deleted the rating distribution of the last time point from the restaurants from the yelp database. We then executed our algorithm on the remaining data, and we predicted the future rating distribution according to Eq. 5. Comparing the predicted ratings with the real (previously deleted) data, two products behaved noticeable among the others. These two products show huge difference to the predicted data, thus, very likely indicating anomalous behavior at the last point in time.

Figure 14 illustrates our observations. In each part of the diagram, the left distribution shows our prediction using the base behavior, while the right distribution presents the observed rating distribution. Clearly, both products show distributions which cannot be explained by the base behavior. The high increase in the number of low rating scores gives a strong indication for anomalous behavior.

Considering the first restaurant, the reviews at the last time point well explain the anomaly. Most of the reviewers comment ... *easily the worst bbq I've ever had...* or ...*I was so hoping to find a good barbecue restaurant near our house. Unfortunately, xxx BBQ just misses the mark...*, thus, indicating that the expectations of a previously high rated restaurant (cf. predicted rating distribution) have been missed. Even more noticeable are the reviews of the second restaurant, where most of the customers criticize the reduced quality using comments as ... *"What a come down! I've been to the xxx quite a few times and have always had a pretty good experience but not this last time ...*. This result gives high indication that the quality of the restaurant has *suddenly* dropped, e.g., due to a change of the chef.

With our method, we can spot these recently occurring anomalies, which can then be used to better inform the customers about the new characteristics of the product/service or to resolve the newly occurring problems from the salesman's perspective.

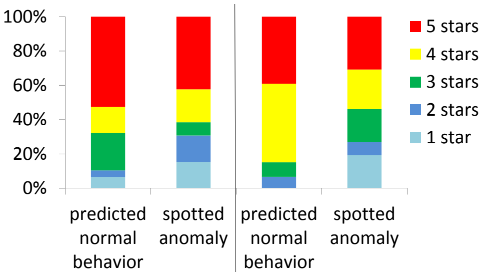

Figure 14: Spotting anomalies via prediction. Restaurants from the Yelp data.

6. CONCLUSION

We presented the RLA method for the temporal analysis of rating distributions. Our method can be used to detect the potentially evolving base behavior of users as well as to spot anomalous points in time where the rating distribution differs from this base behavior. Since our method exploits the temporal characteristics of the data, it further allows to predict the rating distributions of future points in time, which subsequently allows to detect newly occurring anomalies. The idea of our approach is to model the observed data as a mixture of a latent, autoregressive model representing the base behavior with a sparse anomaly signal. Based on this generative process, we derived an efficient algorithm which invokes a sequence of quadratic programs. In our experimental study, we demonstrated the strengths of our RLA method and we presented interesting discoveries found in rating data from Amazon, Yelp, and TripAdvisor.

As future work, we plan to extend our model to higher-order autoregressive processes and we want to investigate how to directly incorporate textual data in our method.

Acknowledgments.

Stephan Günnemann has been supported by a fellowship within the postdoc-program of the German Academic Exchange Service (DAAD). This material is based upon work supported by the National Science Foundation under Grant No. CNS-1314632. Research was also sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. Additional funding was provided by the U.S. Army Research Office (ARO) and Defense Advanced Research Projects Agency (DARPA) under Contract Number W911NF-11-C-0088.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, DARPA, or other funding parties. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

7. REFERENCES

[1] C. C. Aggarwal. *Outlier Analysis*. Springer, 2013.

[2] A. Agresti. *Analysis of ordinal categorical data*. Wiley, 2010.

[3] S. Baccianella, A. Esuli, and F. Sebastiani. Multi-facet rating of product reviews. In *Advances in Information Retrieval*, pages 461–472. Springer, 2009.

[4] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons. Algorithms and

applications for approximate nonnegative matrix factorization. *Computational Statistics & Data Analysis*, 52(1):155–173, 2007.

[5] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2007.

[6] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

[7] E. J. Candès, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted l1 minimization. *Journal of Fourier Analysis and Applications*, 14(5-6):877–905, 2008.

[8] A. L. Chistov and D. Y. Grigor'ev. Complexity of quantifier elimination in the theory of algebraically closed fields. In *Mathematical Foundations of Computer Science 1984*, pages 17–31. Springer, 1984.

[9] C. Croux and K. Joossens. Robust estimation of the vector autoregressive model by a least trimmed squares procedure. In *COMPSTAT*, pages 489–501. Springer, 2008.

[10] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4):411–430, 2000.

[11] M. Jamali, G. Haffari, and M. Ester. Modeling the temporal dynamics of social rating networks using bidirectional effects of social relations and rating patterns. In *WWW*, pages 527–536, 2011.

[12] M. Jamali, T. Huang, and M. Ester. A generalized stochastic block model for recommendation in social rating networks. In *RecSys*, pages 53–60, 2011.

[13] I. Jolliffe. *Principal component analysis*. Wiley Online Library, 2005.

[14] Y. Kawahara and M. Sugiyama. Change-point detection in time-series data by direct density-ratio estimation. In *SDM*, pages 389–400, 2009.

[15] N. Koenigstein, G. Dror, and Y. Koren. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *RecSys*, pages 165–172, 2011.

[16] L. I. Kuncheva. Change detection in streaming multivariate data using likelihood detectors. *IEEE Trans. Knowl. Data Eng.*, 25(5):1175–1180, 2013.

[17] K. Lerman. Dynamics of a collaborative rating system. In *WebKDD/SNA-KDD*, pages 77–96, 2007.

[18] R. B. Litterman. Forecasting with bayesian vector autoregressions. *Journal of Business & Economic Statistics*, 4(1):25–38, 1986.

[19] Y. Liu, X. Yu, A. An, and X. Huang. Riding the tide of sentiment change: sentiment analysis with evolving online reviews. *World Wide Web*, pages 1–20, 2013.

[20] H. Lütkepohl. *New introduction to multiple time series analysis*. Cambridge University Press, 2005.

[21] T. C. Mills. *Time series techniques for economists*. Cambridge University Press, 1991.

[22] S.-H. Min and I. Han. Detection of the customer time-variant pattern for improving recommender systems. *Expert Systems with Applications*, 28(2):189–199, 2005.

[23] S. Mukherjee, G. Basu, and S. Joshi. Incorporating author preference in sentiment rating prediction of reviews. In *WWW*, pages 47–48, 2013.

[24] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

[25] J.-A. Ting, E. Theodorou, and S. Schaal. Learning an outlier-robust kalman filter. In *ECML*, pages 748–756, 2007.

[26] L. Xiong, X. Chen, and J. G. Schneider. Direct robust matrix factorizatoin for anomaly detection. In *ICDM*, pages 844–853, 2011.

APPENDIX

Proof of Theorem 1. Let $\mathbf{p}$ and w be given. Rewriting the objective function $f(...)$, it becomes apparent that we only have to deal with (a) a constant independent of the variables $\mathbf{a}^{(t)}, \mathbf{b}, \mathbf{y}$, which, thus, does not affect their optimal values, and (b) terms $c \cdot u_i$, $c \cdot u_i^2$, and $c \cdot u_i \cdot u_j$, where u_i and u_j are one of the variables $a_d^{(t)}$, b_d, or y_d, and c is a constant depending on the choice which variables are considered. Carefully isolating these constants, leads to the vector $\mathbf{c}^\bullet \in \mathbb{R}^{(T+3) \cdot D}$ with

$$c_i^\bullet = \begin{cases} -2 \cdot x_d^{(t)} \cdot p_t & \text{if } D < i \leq (T+1) \cdot D \\ \sum\limits_{t=1}^{T} 2 \cdot x_d^{(t)} \cdot (p_t - 1) & \text{if } (T+2) \cdot D < i \leq (T+3) \cdot D \\ 0 & \text{else} \end{cases}$$

where we used $t := \left\lceil \frac{i}{D} \right\rceil - 1$ and $d := 1 + [(i-1) \bmod D]$, and to the matrix $\mathbf{Q}^\bullet$ whose diagonal entries are

$$Q_{i,i}^\bullet = \begin{cases} 2 \cdot w^2 & \text{if } 1 \leq i \leq D \\ 2 \cdot p_t^2 + 2 + 2 \cdot w^2 & \text{if } D < i \leq T \cdot D \\ 2 \cdot p_t^2 + 2 & \text{if } T \cdot D < i \leq (T+1) \cdot D \\ 2 \cdot T \cdot (1-w)^2 & \text{if } (T+1) \cdot D < i \leq (T+2) \cdot D \\ \sum\limits_{t=1}^{T} 2 \cdot D \cdot (1-p_t)^2 & \text{if } (T+2) \cdot D < i \leq (T+3) \cdot D \end{cases}$$

and off-diagonal entries $Q_{j,i}^\bullet = Q_{i,j}^\bullet$ (with $i < j$) and

$$Q_{i,j}^\bullet = \begin{cases} -2 \cdot w & \text{if } 1 \leq i \leq T \cdot D \wedge j = i + D \\ -2 \cdot w^2 + 2w & \text{if } 1 \leq i \leq D \wedge j = d + (T+1) \cdot D \\ -2 \cdot (1-w)^2 & \text{if } D < i \leq (T+1) \cdot D \wedge j = d + (T+1) \cdot D \\ 2 \cdot (p_t - p_t^2) & \text{if } D < i \leq (T+1) \cdot D \wedge j = d + (T+2) \cdot D \\ 0 & \text{else} \end{cases}$$

The matrix $\mathbf{Q}^\bullet$ is sparse because only $7 \cdot T \cdot D + 5 \cdot D$ elements are not zero. Since the set $\mathcal{C}_D$ used in problem P1, adds only linear constraints to the variables $\mathbf{a}^{(t)}, \mathbf{b}, \mathbf{y}$, it is easy to derive matrices $\mathbf{A}^\bullet$, $\mathbf{d}^\bullet$ describing these constraints. □

Proof of Theorem 2. Rewriting the objective function shows that it only involves terms $c \cdot v_i$ and $c \cdot v_i^2$, where v_i is one of the variables p_t or w, and c is a constant depending on the choice which variable is considered. Isolating these constants, leads to the vector $\mathbf{c}^\circ$ with

$$c_i^\circ = \begin{cases} \sum\limits_{d=1}^{D} -(a_d^{(i)} - y_d)(x_d^{(i)} - y_d) & \text{if } 1 \leq i \leq T \\ \sum\limits_{t=1}^{T} \sum\limits_{d=1}^{D} -(a_d^{(t)} - b_d)(a_d^{(t-1)} - b_d) & \text{if } i = T+1 \end{cases}$$

and to the diagonal matrix $\mathbf{Q}^\circ = diag(q_1^\circ, \ldots, q_{T+1}^\circ)$ with

$$q_i^\circ = \begin{cases} \sum\limits_{d=1}^{D} 2 \cdot (a_d^{(i)} - y_d)^2 & \text{if } 1 \leq i \leq T \\ \sum\limits_{t=1}^{T} \sum\limits_{d=1}^{D} 2 \cdot (a_d^{(t-1)} - b_d)^2 & \text{if } i = T+1 \end{cases}$$

Obviously, $\mathbf{Q}^\circ$ is a sparse matrix. Since the L_0 norm used in the original problem definition has been replaced with the linear constraint of Equation 9, it is again easy to derive matrices $\mathbf{A}^\circ$, $\mathbf{d}^\circ$ describing these constraints. □