The Latent Behavior Space - A Vector Space for Time-Series Data

Jan Reubold [*1] **Stephan Escher** [*1] **Thorsten Strufe** [1]

Abstract

The timing and temporal order are two characteristic properties that are frequently omitted in machine learning approaches, but carry crucial information. Their consideration is currently limited to algorithms that are specialized to sequential data, but it takes a projection into a vector space to employ the wealth of ML algorithms that are known and understood. Projections inevitably cause a loss of detail. A naïve application of *bag-of-words*, as a prominent example, utilizes neither order nor timing of events. In this paper we introduce a projection strategy that retains order and timings. It identifies the latent space that the generating processes underlying the time series are spanning.

1. Introduction

Consecutive measurements of processes naturally yield sequences of presumably interdependent observations, and time-series hence have become a very popular representation of data. Therefore, we see an increase in attempts to extract patterns and learn algorithms on time-series data in various domains. These include recommendations for locations, music, and products of potential interest (Figueiredo et al., 2016), as well as the detection of social bots (Viswanath et al., 2014), the forecasting of financial data (Cao & Tay, 2003), and the mining of electronic health records (Jensen et al., 2012). All these applications rely on approaches able to process time-series data.

The current trend is directed at specializing algorithms to specific types of time-series data (Figueiredo et al., 2016; Saeedi et al., 2016; Beal et al., 2001; Fine et al., 1998). This, however, prevents leveraging the benefits of the broad majority of the known and understood ML algorithms, as they rely on projections onto fixed dimensions. Additionally, a projection reduces the problem complexity and allows for the augmentation of context information, like, e.g., the age and gender of an observed subject. However, common projections retain only highly reduced information of the input data, which unnecessarily limits the potential quality and performance of the ML algorithms applied to them (Wallach, 2006). An example is the naïve utilization of a *bag-of-words* approach. It projects the data into a vector space by simply mapping the frequency of each observation to the different dimensions. Thus, the information retained by the projection hinges on the design of the feature space the *bag-of-words* is applied to, i.e. the representation of observed time-series data.

A common approach for constructing a latent vector space factoring in sequential information is to utilize *n-grams*. *N-gram* methods represent data by the frequency count of segments of length $n+1$ within the time-series data. Spanning the latent space by the set of unique *n-grams* would incorporate sequential information into the process, but also hold obvious drawbacks. For one, such a representation could potentially result in an explosion of the vector space dimensionality. Also, the lack of a reasonable abstraction mechanism potentially increases the information loss induced by the projection.

An adjustment to this approach is to mine patterns from the *n-gram* representations of time-series data (Cadez et al., 2000; Wallach, 2006). This mechanism provides a level of abstraction to mitigate the mentioned drawbacks. Figueiredo et al. (2016) propose an enhanced approach that applies a static segment identification before mining underlying patterns. By splitting the data into smaller chunks to then model their dynamics results in a more accurate representation of information contained in time-series data.

In this paper, we bridge the gap between specialized sequential algorithms and approaches on fixed dimensional input data. We claim that applying these algorithms to identify the axes of a latent feature space for the utilization of approaches on fixed dimensional input data can significantly improve the performance of these approaches. The proposed specialized algorithm identifies segments in categorical-valued time-series data (e.g. click traces), by approximating underlying latent processes. It automatically splits the time-series into segments of arbitrary length while modeling the processes. The result is a latent vec-

[*]Equal contribution [1]Technical University of Dresden, Saxony, Germany. Correspondence to: Jan Reubold <jan.reubold@tu-dresden.de>.

Preliminary work. Under review by the Time-Series Workshop @ICML 2017. Do not distribute.

tor space, spanned by the detected processes. This projection yields the benefits of vector space representations and maintains timing and sequential information. We finally sketch an application of our projection for *social bot detection*, by outlining an evaluation strategy that measures the impact of the proposed *Latent Behavior Space (LBS)*. Thus, our contributions in this paper are as follows, we propose

(i) a standardized scheme to project time-series data into a suitable vector space,

(ii) a segmentation algorithm on categorical-valued time-series data incorporating timing and temporal order,

(iii) an evaluation strategy to measure the performance of *social bot detection* methods.

2. Latent Behavior Space

The core component of the *LBS* is its approach to identify the latent dimensions, i.e. the latent processes underlying time-series data. In the following we describe the segmentation algorithm in more detail before illustrating the *LBS*.

2.1. Behavior Axes

The segmentation algorithm extends previous work (Reubold et al., 2017) by integrating a click-duration model. It takes as input a set of M time-series $\{\mathbf{x_i}\}_{i=1}^{M}$ with elements $x_{ij} \in Y$ with $j \in [1, l_i]$ and l_i denoting the length of $\mathbf{x_i}$. Y denotes the set of possible observations. The algorithm consists of a mixture model. In the remainder of this work we refer to the corresponding mixture components as super states $c \in C$.

Super states are a representation of repeating patterns which we learn from our observational data. For example an online user can be observed by a series of click traces. Each such state belongs to something the user wants to do, i.e. a super state. For example checking their mail or replying to someone on social media. With the context of previous observations and intentions we can assign the current event a probability of belonging to a certain super state. The vector space spanned by the frequency of observed user intentions, i.e. super states, then allows us to represent time series' as vectors. This vector partially preserves temporal information, yet allows the addition of further dimensions, such as the gender or age of an online user, and the application of common ML techniques. We will now describe this process formally.

Mixture Model. In order to extract patterns from within time-series data, the algorithm learns a set of super states C upper bounded by N. Each super state c is comprised of a set of internal states representing the observations in Y. For each observation x_{ij} within time-series $\mathbf{x_i}$ the corresponding super state is denoted by a latent parameter z_{ij}. To not clutter the notation, we omit subscripts i and j whenever context allows.

The transitions within- and transitions between super states are modeled by Multinomial distributions (`Mu`). Their prior distributions consist of hierarchically arranged Dirichlet distributions (*Dir*). In the following, we describe the transition model between super states, the transition model within super states, and conclude with a summary of the generative process.

Super State Transitions. To model the dependencies between successive super states we apply a hierarchical model. The design expresses our prior belief that the conditioned transition probabilities of super states are governed by their prior probabilities. Therefore, the top layer encodes the importance of super states and is represented by a N-dimensional Dirichlet distribution

$$\beta|\gamma \sim Dir(\gamma), \tag{1}$$

where γ represents the base distribution. The bottom layer is governed by β and represents the transition probability between super states conditioned on the active super state z_{ij-1}, $p(z_{ij}|z_{ij-1} = c)$,

$$\pi_c|\alpha, \beta \sim Dir(\alpha\beta + \rho\mathbb{1}_c), \tag{2}$$

where $\Pi \triangleq \{\pi_c\}_{c \in C}$, ρ represents a bias towards self-transitions, $\mathbb{1}$ the indicator function, and α, γ base distributions. The bias towards self-transitions represents our prior belief that a successive observation has an increased probability to be generated from the same super state as its predecessor. Technically, it acts as a countermeasure to fast switching between redundant states in the hierarchical model and, therefore, a possible decrease in prediction performance (Fox et al., 2011).

Super States. Similar to MCs, super states are expressed by an initial-state distribution θ^I and a transition distribution θ^T. We augment the latter by an additional distribution and internal end-state E. Internal end-states provide a measure of the super state duration by leveraging the observations recorded at the end of segments. Finally, the additional distribution provides a means to measure the duration between successive observations in a segment. Due to these additions our model can capture both, timings and temporal order.

For ease of notation, the initial- and transition distribution of a super state c are denoted by $\theta_c \triangleq \{\theta_c^I, \theta_c^T\}$. The prior for the initial-state distribution, θ^I, is modeled by a *Dir* representing $p(x|\theta^I)$,

$$\theta_c^I|\lambda_c^I \sim Dir(\lambda_c^I), \tag{3}$$

where λ denotes the base distribution. The transition model of the internal state θ^T is expressed similar to the super state transition model. Here, G_c denotes the importance of the internal states of super state c, $p(x|c)$,

$$G_c|\psi_c \sim Dir(\psi_c), \tag{4}$$

with ψ_c denoting a base distribution. The transition distribution conditioned on the active super state and previous internal state, $p(x_{ij}|z_{ij} = c, x_{ij-1} = s)$ with $s \in Y$, is then

$$\theta^T_{cs}|\lambda^T_c, G_c \sim Dir\left(\lambda^T_c, G_c\right), \tag{5}$$

where $\theta^T_c \triangleq \{\theta^T_{cs}\}_{s \in S_c}$. With the addition of the internal end-state Eq. 2 has to be adjusted,

$$\pi_c|\alpha, \beta \sim Dir\left(\alpha\beta + (1-\tau)\,\rho\mathbb{1}_c\right), \tag{6}$$

with τ denoting an indicator function which is 1 if $x_{ij-1} \in \{\emptyset, E\}$ and 0 otherwise. This adjustment factors in the information of a certain end of the current segment ($\tau = 1$).

For the internal state duration model, we extend the conditioned transition distribution θ^T by one-dimensional Gaussian mixture models (GMMs). Each Gaussian mixture component encodes the duration distribution of a state condition on the successive state. Therefore, θ^T is extended to also depend on the time spent in the current state. The prior for the parameters of the mixture model are sampled as

$$\begin{aligned} \mathbf{\Sigma}_{cs} &\sim \mathtt{IW}_{\upsilon_0}\left(\Lambda_0^{-1}\right) \\ \mu_{cs} &\sim \mathcal{N}\left(\mu_0, \Sigma_{cs}/\kappa_0\right), \end{aligned} \tag{7}$$

where $\mathtt{IW}$ is the Inverse-Wishart distributions with $H = \{\Lambda^{-1}, \upsilon_0, \mu_0, \kappa_0\}$ its parameters. $\mathcal{N}$ denotes the Normal distribution.

Generative Model. Finally, we can define the generative process of our model. An element of a time-series is generated by one of the super states which is recorded by the layer of latent parameters $\mathbf{z}$. If $x_{ij-1} \notin \{\emptyset, E\}$, $z_{ij} = z_{ij-1}$, otherwise,

$$z_{ij}|\widetilde{\pi}_{z_{ij-1}} \sim \mathtt{Mu}\left(\widetilde{\pi}_{z_{ij-1}}\right) \tag{8}$$

where $\widetilde{\pi}$ is similar to π, but accounting only for the first element of a segment, factoring out state transitions within super states. Additionally, we keep track of the first element of segments by recording their indexes $[i,j] \in \widetilde{\mathbf{z}}$.

Given the corresponding super state z_{ij} and the previous internal state x_{ij-1} with t_{ij-1}, the successive state is sampled by

$$x_{ij}|z_{ij}, x_{ij-1}, t_{ij-1} \sim \mathtt{Mu}\left(\theta_{z_{ij}t_{ij-1}}\left[x_{ij-1}\right]\right), \tag{9}$$

where θ either corresponds to θ^I, if x represents the first element of a segment, $x_{ij-1} \in \{\emptyset, E\}$, or θ^T otherwise.

Finally, the state duration t_{ij} is sampled by

$$t_{ij}|z_{ij}, x_{ij} \sim \mathcal{N}(\mu_{z_{ij}x_{ij}}, \mathbf{\Sigma}_{z_{ij}x_{ij}}). \tag{10}$$

The approach leverages first-order dependencies of super- and internal state to identify common patterns within time-series data. Information on the truncated Gibbs Sampler and further details can be found in (Reubold et al., 2017). With the obtained set of behavior patterns, we can now span the latent behavior space.

2.2. Latent Space

In this section we define a suitable latent space for time-series data. It accounts for the sequential information inherent in the data while allowing to apply standard point based vector space methods, e.g. SVM, kNN, and other well known approaches.

Consider the previous example of users surfing on the Web. Each user is represented by a set of time-series, i.e. click-traces. Given the corresponding *LBS*, a user is represented as a point in this latent space encoding his intentions instead of the exact click sequences.

Let us assume $\widetilde{C}$ is the set of relevant/active super states, $|\widetilde{C}| \leq N$. Then, the vector space V is spanned by these super states. Each axis represents one of the patterns in $\widetilde{C}$. Thus, a set of time-series $\mathbf{X} = \{\mathbf{x}_b\}_{b \in B}$ with cardinality B, e.g. the set of time-series representing a user, can be expressed as a point $v \in \mathbb{R}^D$,

$$v = \phi_{\mathbf{z}}(\mathbf{X}), \tag{11}$$

where $\phi_{\mathbf{z}}$ denotes the projection as a function of the latent super state assignments $\mathbf{z}$. Depending on the application we provide two variants of ϕ, one representing the occurrences of super states ϕ^O and a second representing time spent in each of the super states ϕ^D,

$$\begin{aligned} \phi^O_{\mathbf{z}}(\mathbf{X}) &\triangleq \frac{\sum_{i,j \in \widetilde{\mathbf{z}}} \mathbb{1}_{z_{ij}}}{|\widetilde{\mathbf{z}}|}, \\ \phi^D_{\mathbf{z}}(\mathbf{X}) &\triangleq \frac{\sum_{1 \leq i \leq B} \sum_{1 \leq j \leq l_i} \mathbb{1}_{z_{ij}} t_{ij}}{\sum_{1 \leq i \leq B} \sum_{1 \leq j \leq l_i} t_{ij}}. \end{aligned} \tag{12}$$

Compared to approaches representing the data by naïve bag-of-words projections, such a representation retains significantly more of the original information contained in time-series data.

3. Application

In the following, we outline a suitable evaluation strategy. In experiments, we replace the individually devised projection strategies proposed in papers by the *LBS*. This setup allows to measure the impact of the *LBS* in the context of applications utilizing behavioral time-series data.

Social Bot Detection. One possible application of our proposed algorithm is in the area of social bot detection. A social bot is an automation software created to control an Online Social Network (OSN) account (fake or compromised profile) and tries to pose as a human. To improve their range of influence they can be connected to so called 'social bot networks' (Boshmaf et al., 2013). The intentions behind social bots range from identity theft to influencing the society (hype or denounce people, organisations, political discussions, etc.), to the distribution of malicious content (e.g. spam, phishing, malware). Overall they are created for harvesting and/or distributing information.

Existing countermeasures are widespread. From prevention, like fast-response captchas, to social graph based sybil detection (fake accounts) (Cao et al., 2012; Boshmaf et al., 2015) to crowd based- and (Beutel et al., 2013; Cao et al., 2014; Li et al., 2016) behavior based detection methods (Wang et al., 2013; Viswanath et al., 2014). The latter try to differentiate between sybils, cyborgs (compromised accounts) and humans by leveraging behavioral observations. According to the intention, sophistication and structure (cyborg or sybil), social bots exhibit different behavior patterns distinct to real users, like, e.g., frequent liking or posting of content.

Wang et al. (2013) developed a semi-supervised detection algorithm based on a clickstream similarity graph for sybil detection. They map clickstreams (click-traces and timings) to a similarity graph, where clickstreams (vertices) are connected using weighted edges that capture pairwise similarity. Finally they apply graph partitioning to identify clusters that represent specific click patterns.

Viswanath et al. (2014) proposed an unsupervised anomaly detection technique. They project the user behavior traces (timings, temporal order, and combined features) using the *bag-of-words* approach. Using the transformed data they extract a latent vector space identified by Principal Component Analysis (PCA).

Hypothesis. In order to evaluate the impact of the *LBS* we propose a modification to their methodologies.

First, instead of constructing a graph with user as vertices and distances between them as edge weights (Wang et al., 2013), we suggest to use the *LBS*. Here, users are represented by a weighted set of shown intentions. Therefore, the euclidean distance represents a reasonable similarity measure. Our hypothesis is that the level of abstraction gained by this projection yields more reasonable clusters of similar user behavior.

Second, instead of using a naïve *bag-of-words* projection, as proposed by Viswanath et al. (2014), we suggest to use the *LBS*. Our hypothesis is that this procedure retains crucial sequential information which otherwise would be lost.

Data Set. Evaluating social bot detection algorithms pose a challenging task. Approaches making use of labeled datasets are biased, because only known attack behaviors are labeled as such. In addition to real world dataset, we suggest to enrich the data. By creating theoretically attacker models, one can build generating processes to emulate such attacks. These attacker models depend on parameters like attack intention, bot complexity or the structure of the targeted OSN, in addtion to a benefit-cost analysis. For the design of these models we include related work knowledge (Boshmaf et al., 2013; Thomas & Nicol, 2010).

An example of such a model could be a bot that co-acts on a real client, like the koobface botnet (Thomas & Nicol, 2010). Such a bot could have the ability to analyze and adapt the specific user behavior. While less beneficial for attack intentions like spam or harvesting information (because of the increased cost and decreased influence factor) these bots can potentially be of high value for e.g. influencing political discussions or hyping different products over a longer period of time.

Enriching real-world data sets with additional artificial bot traces potentially yield a better understanding of the reach and performance of proposed detection approaches.

4. Summmary and Outlook

In this paper, we proposed an approach to construct latent vector spaces for time-series data. Therefore, we designed a model to automatically split time-series data into segments that represent identified processes. It allowed to integrate temporal information into the modeling phase while providing a natural abstraction mechanism. Our approach presents a trade-off between modeling detail and data abstraction. Different trade-off strategies result in different modeling approaches. While our approach presents a solid base algorithm (timing and temporal order), further research could lead to a family of time-series projection strategies. In our opinion, such a development will result in a significant advancement in the field.

Finally, we proposed an evaluation strategy that allows to decrease the bias in *social bot detection* evaluations. An enriched data set allows for evaluating the detection performance on real-world data as well as testing against unknown attack strategies.

Acknowledgements

This research has been funded in parts by the German Science Foundation DFG under grant GRK/1907 and by the German Federal Ministry of Education and Science BMBF under grant QQM/01LSA1503C.

References

Beal, Matthew J, Ghahramani, Zoubin, and Rasmussen, Carl E. The infinite hidden markov model. In *Advances in neural information processing systems*, pp. 577–584, 2001.

Beutel, Alex, Xu, Wanhong, Guruswami, Venkatesan, Palow, Christopher, and Faloutsos, Christos. Copycatch: stopping group attacks by spotting lockstep behavior in social networks. In *Proceedings of the 22nd international conference on World Wide Web*, pp. 119–130. ACM, 2013.

Boshmaf, Yazan, Muslukhov, Ildar, Beznosov, Konstantin, and Ripeanu, Matei. Design and analysis of a social botnet. *Computer Networks*, 57(2):556–578, 2013.

Boshmaf, Yazan, Logothetis, Dionysios, Siganos, Georgos, Lería, Jorge, Lorenzo, Jose, Ripeanu, Matei, and Beznosov, Konstantin. Integro: Leveraging victim prediction for robust fake account detection in osns. In *NDSS*, volume 15, pp. 8–11, 2015.

Cadez, Igor, Heckerman, David, Meek, Christopher, Smyth, Padhraic, and White, Steven. Visualization of navigation patterns on a web site using model-based clustering. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 280–284. ACM, 2000.

Cao, Li-Juan and Tay, Francis Eng Hock. Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on neural networks*, 14(6):1506–1518, 2003.

Cao, Qiang, Sirivianos, Michael, Yang, Xiaowei, and Pregueiro, Tiago. Aiding the detection of fake accounts in large scale social online services. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pp. 15–15. USENIX Association, 2012.

Cao, Qiang, Yang, Xiaowei, Yu, Jieqi, and Palow, Christopher. Uncovering large groups of active malicious accounts in online social networks. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pp. 477–488. ACM, 2014.

Figueiredo, Flavio, Ribeiro, Bruno, Almeida, Jussara M, and Faloutsos, Christos. Tribeflow: Mining & predicting user trajectories. In *Proceedings of the 25th International Conference on World Wide Web*, pp. 695–706. International World Wide Web Conferences Steering Committee, 2016.

Fine, Shai, Singer, Yoram, and Tishby, Naftali. The hierarchical hidden markov model: Analysis and applications. *Machine learning*, 32(1):41–62, 1998.

Fox, Emily B, Sudderth, Erik B, Jordan, Michael I, and Willsky, Alan S. A sticky hdp-hmm with application to speaker diarization. *The Annals of Applied Statistics*, pp. 1020–1056, 2011.

Jensen, Peter B, Jensen, Lars J, and Brunak, Søren. Mining electronic health records: towards better research applications and clinical care. *Nature Reviews Genetics*, 13 (6):395–405, 2012.

Li, Yixuan, Martinez, Oscar, Chen, Xing, Li, Yi, and Hopcroft, John E. In a world that counts: Clustering and detecting fake social engagement at scale. In *Proceedings of the 25th International Conference on World Wide Web*, pp. 111–120. International World Wide Web Conferences Steering Committee, 2016.

Reubold, Jan, Strufe, Thorsten, and Brefeld, Ulf. Infinite mixture model of markov chains. In *arXiv*, 2017.

Saeedi, Ardavan, Hoffman, Matthew, Johnson, Matthew, and Adams, Ryan. The segmented ihmm: A simple, efficient hierarchical infinite hmm. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 2682–2691, 2016.

Thomas, Kurt and Nicol, David M. The koobface botnet and the rise of social malware. In *Malicious and Unwanted Software (MALWARE), 2010 5th International Conference on*, pp. 63–70. IEEE, 2010.

Viswanath, Bimal, Bashir, Muhammad Ahmad, Crovella, Mark, Guha, Saikat, Gummadi, Krishna P, Krishnamurthy, Balachander, and Mislove, Alan. Towards detecting anomalous user behavior in online social networks. In *Usenix Security*, volume 14, 2014.

Wallach, Hanna M. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pp. 977–984. ACM, 2006.

Wang, Gang, Konolige, Tristan, Wilson, Christo, Wang, Xiao, Zheng, Haitao, and Zhao, Ben Y. You are how you click: Clickstream analysis for sybil detection. In *Usenix Security*, volume 14, 2013.