SDM-EDA 2014
The First Workshop on Exploratory Data Analysis

http://cda.gatech.edu/sdm14eda

April 26, 2014, Philadelphia, PA

In conjunction with
14th SIAM International Conference on Data Mining (SDM 2014)

Organizers

Remco Chang, Tufts University
Jaegul Choo, Georgia Institute of Technology
Zhicheng Liu, Adobe Research
Haesun Park, Georgia Institute of Technology

Objectives and Topics of Interest

The data mining and machine learning research often tackles the problem in a fully automated manner while the other side of research community, such as human-computer interaction and information visualization, does not fully take advantage of the recent advancements in state-of-the-art data mining techniques.

The primary goal of the workshop is to fill this gap by bringing together researchers from both sides. The workshop should provide an opportunity to discuss and explore ways to harmonize the power of data mining techniques and human-in-the-loop exploratory nature of data analysis.

Specifically, the workshop will investigate these challenges and discuss promising directions.

- *Interactive data mining algorithms*. Most data mining algorithms work in a fully automated manner. How can these algorithms provide user interactions so that they can properly incorporate the user intention? How can they provide meaningful interpretation about the algorithm results?
- *Visualizations for interactive data mining.* One of the most effective ways to facilitate exploratory analysis is visualization of data and the output of data mining algorithms. Data usually contain multiple types of information, and the data mining techniques also involves complex outputs. How can we effectively deliver such information to human via visualization?

- *Visual analytics systems based on data mining techniques.* Integrating visualizations and various interactions that support user intentions into a mature visual analytics system is crucial. How can data mining techniques be effectively integrated into visual analytics systems?
- *Any-time data mining algorithms*. To provide smooth interactions with data mining techniques, it is crucial that they should give the results at any time that humans want. How can we accelerate data mining algorithms and what aspects should we exploit to serve this purpose?

Other than these topics, the workshop will broadly cover other related directions such as

- Demonstrations of interactive data mining
- Data surrogates
- On-line algorithms
- Adaptive stream mining algorithms
- Theoretical/complexity analysis of instant data mining
- Learning from user input for action replication/prediction
- Active learning / mining

Planned Activities

We intend to have four sessions in the workshop (one day). The two sessions in the morning will contain eight invited talks, each 25-minute long. The third session after an hour lunch break will contain three more invited talks, each 25-minute long, followed by quick poster presentations, each 5-minute long. During the break in the morning and the afternoon, workshop audiences will be able to look into posters and interact with poster presenters. All posters are based on contributed submissions. The organizers has reviewed the submissions and selected high-quality submissions for poster presentation.

Finally, the last session will be panel discussion. The panel discussion will focus on challenges in integrating data mining/machine learning and InfoVis/HCI for big data. The panelists will be recruited from experts in the field, including the invited speakers. One or more of the organizers will moderate the panel. Questions will be taken from the audience and the moderator and panelists will be free to contribute additional questions.

Tentative Schedule (finalized schedule is at http://cda.gatech.edu/sdm14eda)

	Speaker(Affiliation)	Talk Title
Session 1: Invited Talks (8:30am - 10:15am)	Polo Chau, Georgia Institute of Technology	Scalable, Interactive, and Comprehensible Tools for Data Analytics
	Carlos Scheidegger, AT&T Labs	RCloud and the gap between EDA, collaboration, and deployment
	Lisa Singh, Georgetown University	Improving Data Exploration Through the Use of Interactive Workflows and Visual Analytics
	Claudio T. Silva, Polytechnic Institute of NYU	To be announced
Break (10:15am - 10:30am)		
Session 2: Invited Talks (10:30am - 12:30pm)	Eytan Adar, University of Michigan	Social and Individual Biases in Visual Analytics
	Jieping Ye, Arizona State University	Multi-Source Feature Learning
	John Stasko, Georgia Institute of Technology	The Value of Visualization for Exploring and Understanding Data
	Fei Sha, University of Southern California	To be announced
Lunch (12:30pm - 1:30pm)		
Session 3: Invited Talks (1:30pm - 3:10pm)	Klaus Mueller, Stony Brook University	Model-Driven Visual Analytics
	Srinivasan Parthasarathy, Ohio State University	On the Roles of Visualization within Data Analytics
	Hadley Wickham, Rice University	Expressing your EDA in R
	Tamara Munzner, University of British Columbia	Dimensionality Reduction from Three Angles
Break (3:10pm – 3:30pm))		
Poster Presentation / Panel Discussion (3:30pm - 5:00pm)	Panelists: To be announced	- What benefits an exploratory analysis approach can have compared to completely automated approaches? - In what problems, can exploratory analysis be helpful? In what problems, is it not? - What are the challenges in exploratory analysis of big data? - How much important is the interactive visualization in exploratory analysis? - What are the challenges in integrating exploratory analysis approaches, e.g., interactive visualization, with automated approaches? - How can we increase the collaboration between data mining/machine learning and InfoVis/HCI communities?

Invited Talk Information (in an alphabetical order of speakers' last name)

Social and Individual Biases in Visual Analytics
Speaker: **Eytan Adar**, University of Michigan
Abstract: We are all sensitive to bias in our interpretation of data. Even with awareness of the various data-driven, perceptual, cognitive processes, and social mechanisms that can bias our understanding we regularly fall into "traps" that lead to significant mistakes in EDA tasks. In this talk I'll cover some of the recent work in my group on the negative effect of certain social processes (e.g., social proof) and individual heuristics (e.g., primacy and recency effects) on visualization interpretation. As a consequence of these findings we have begun to develop a number of techniques to display "alternative" versions of the data. By providing these alternatives in a systematic way, we can provide end-users with the tools to more accurately interpret data and model uncertainty. Our hope is that techniques such as ours can address the biasing effects we regularly experience and to help create a more advanced class of EDA tools.

Scalable, Interactive, and Comprehensible Tools for Data Analytics
Speaker: **Polo Chau**, Georgia Institute of Technology
Abstract: Massive datasets now arise in virtually all domains. Yet, making sense of these data remains a fundamental challenge. At the Polo Club of Data Science, we are innovating at the intersection of Data Mining and Human-Computer Interaction (HCI), to combine the best from both worlds to create scalable, interactive tools for making sense of graphs with billions of nodes and edges. I will briefly describe some of our latest works, both on-going and published, that aim to tame big data through scalable algorithms, interactive visualization, and comprehensible models (machine learning/data mining) that users can more easily understand and work with.

Model-Driven Visual Analytics
Speaker: **Klaus Mueller**, Stony Brook University
Abstract: The growth of digital data is tremendous. We can now afford to explore very complex relationships with many variables playing a part. But for this we need powerful and interactive tools that allow us to be creative, to sculpt this intricate insight formulated as models from the large raw block of data. High-quality visual feedback plays a decisive role here. In this talk, I will first provide some fundamental insight into the peculiarities of high-dimensional data spaces and also discuss our recent software framework, called "The ND-Scope", which incorporates various facilities for high-dimensional data exploration. A frequent outcome of such data explorations is a model. Models can summarize the original big data into a concise representation. However, the process of learning models from raw data typically requires a substantial amount of user input during the model initialization phase. To this end, I will present an extension of our visualization system which greatly reduces the load on the users and makes the process of model initialization and refinement significantly more efficient, problem-driven, and engaging.

Dimensionality Reduction from Three Angles
Speaker: **Tamara Munzner**, University of British Columbia
Abstract: I will present three projects that attack the problem of dimensionality reduction (DR) in visualization from different methodological angles of attack, in order to answer different kinds of questions. First, can we design better DR algorithms? Glimmer is a multilevel multidimensional scaling (MDS) algorithm that exploits the GPU. Second, can we build a DR system for real people? DimStiller is a toolkit for DR that provides local and global guidance to users who may not be experts in the mathematics of high-dimensional data analysis, in hopes of "DR for the rest of us". Third, how should we show people DR results? A data study, where a small number of people make judgments about a large number of datasets rather than vice versa as with a typical user study, produced a taxonomy of visual cluster separation factors.

On the Roles of Visualization within Data Analytics
Speaker: **Srinivasan Parthasarathy**, Ohio State University
Abstract: Data Analytics is a process whose goal is to extract interpretable and actionable information from complex (potentially large) data. Visualization can play an important role in this exploratory process. Indeed, a number of important scientific discoveries have ultimately relied on visual confirmation – from Galileo seeing the moons of Jupiter to Gerd Binnig and Heinrich Rohrer seeing atoms on a surface. Visualization can also play an important role in understanding the nature of a problem domain and subsequently the patterns governing the underlying solution space. In this talk I will talk about our vision on the roles visualization can play in the knowledge discovery process. Specifically we will examine the use of visualization:

1. As a mechanism to facilitate exploration of complex datasets.
2. As a means to validate and confirm results obtained from the discovery process.
3. As an approach to understand and lend transparency to the discovery process.

In each case I will attempt to illustrate the roles in the context of specific end applications drawn from the domains of physics of materials, bioinformatics, social network analysis and clinical diagnosis of eye disease. No prior knowledge of data mining, knowledge discovery, visualization or any of these application domains will be assumed.

RCloud and the gap between EDA, collaboration, and deployment
Speaker: **Carlos Scheidegger**, AT&T Labs
Abstract: Consider the emerging role of a data science team within an organization. Different data scientists and statisticians usually work on loosely related problems, and transition fluidly from exploratory data analysis to automated diagnostics and reports, which need to be deployed for wider consumption on the organization. There are two problems with the current practice. First, there is a gap in this workflow: EDA is performed with one set of tools, and automated reports and deployments with another. Second, current EDA environments mostly assume a single-developer perspective, while data scientist teams could greatly benefit from easier sharing of scripts, beyond that which a traditional version control system can provide. In this talk, I will present RCloud, designed to enable collaborative data analysis and deployment on the web. I will discuss the design decisions, tradeoffs and limitations, and the current status of the open-source project.

Improving Data Exploration Through the Use of Interactive Workflows and Visual Analytics
Speaker: **Lisa Singh**, Georgetown University
Abstract: Social scientists and observational scientists have a need to analyze complex network data sets. Some exploratory tasks of interest include community detection, subgraph similarity analysis, and anomaly detection. Even though many algorithms and visualizations have been developed to help users understand results obtained from data mining algorithms, many of the scientists using these tools are still uncomfortable with the process of data exploration and may be unable to effectively explore their data sets as a part of the scientific inquiry process. This talk will discuss the challenges associated with the process of data exploration and describe a number of approaches and tools that attempt to overcome these challenges through the use of interactive workflows and visual analytics.

The Value of Visualization for Exploring and Understanding Data
Speaker: **John Stasko**, Georgia Institute of Technology
Abstract: Investigators have an ever-growing suite of tools available for analyzing and understanding their data. While techniques such as statistical analysis, data mining, and machine learning all have benefits, visualization provides an additional unique set of capabilities. In this talk I identify the particular advantages that visualization brings to data analysis beyond other techniques, and I will describe the situations in which it can be most beneficial.

Expressing your EDA in R
Speaker: **Hadley Wickham**, Rice University
Abstract: There are three main time sinks in any exploratory data analysis:

1. Figuring out what you want to do.
2. Turning a vague goal into a precise set of tasks (i.e. programming).
3. Actually crunching the numbers.

A well-design domain specific language (or DSL) tightly coupled to the problem domain can make all three pieces faster. I'll discuss two DSLs built in R: dplyr for data manipulation, and ggvis for visualisation. These build on my previous work (plyr and ggplot2) improving both expressivity and speed.

Data visualisation and manipulation are key parts of EDA. dplyr implements the most important verbs of data manipulation in a datastore-agnostic fashion, so you can think about and compute with your data in the same way regarldess of whether you're working with a local in-memory data frame or a remote on-disk database. ggvis makes it easy to declaratively describe reactive (interactive and dynamic) web graphics. It combines a declarative syntax based on ggplot2 with functional reactive programming (from shiny) reactive programming model and a declarative JS rendering system (from vega).

Multi-Source Feature Learning
Speaker: **Jieping Ye**, Arizona State University
Abstract: Recent advances in data collection technologies have made it possible to collect a large amount of data for many application domains. Very often, these data come from multiple sources. For instance, in the study of Alzheimer's Disease (AD), different types of measurements such as magnetic resonance imaging (MRI), positron emission tomography (PET), cerebrospinal fluid (CSF), blood test, gene/protein expression data, and genetic data have been collected as they provide complementary information for the diagnosis of AD. A joint learning of multiple data sources is beneficial as different data sources may contain complementary information, while feature selection is critical for learning interpretable models from high-dimensional data. The key challenge in multi-source feature learning is how to effectively integrate information from multiple heterogeneous sources. In this talk, I will present a unified feature learning model for multi-source data.

Workshop Papers (Full manuscripts are shown in the following pages.)

Hierarchical Mixing Linear SVMs based on Rademacher Complexity *

Di Wang †, Xiaoqin Zhang †, Tang Tang †, Mingyu Fan †

Abstract

Support vector machines (SVMs) suffer from the high computational complexity in the classification phase when there are a considerable number of support vectors (SVs). Then it is desirable to design efficient algorithms in the classification phase to deal with the datasets in real-time pattern recognition systems. To this end, we propose a novel classifier called H-MLSVMs (Hierarchical Mixing Linear Support Vector Machines) in this paper, which has a hierarchical structure with a mixing linear SVMs classifier at each node and predicts the label of a sample using only a few hyperplanes. We also give a generalization error bound for the class of local linear SVMs (LLSVMs) based on the Rademacher theory, which makes H-MLSVMs avoid overfitting. The classifier exploits both the advantage of good generalization ability of kernel SVMs (KSVMs) and the advantage of the high speed of linear SVMs (LSVMs) in the classification stage. Experimental results show the high efficiency and the good classification performance of the proposed classifier.

1 Introduction

Support vector machines (SVMs) has enjoyed excellent performance in many areas such as face recognition [1], tracking [2] and so on. The "kernel trick" provides one of the main building blocks of kernel SVMs (KSVMs). However, they suffer from the expensive computational complexity in the classification phase since the solution of KSVMs is parameterized by a set of training samples which are called support vectors (SVs). When the number of SVs is large, KSVMs is time-consuming in the classification.

A good strategy for overcoming the limitation of KSVMs is to construct ensemble based classifiers in the original space, thus avoiding the kernel calculation of the SVs. Classical ensemble based methods include boosting [3] and random forests [4]. But these methods need many linear classifiers to correctly predict the label for linearly inseparable datasets. Fu *et al.* [5] propose a mixture of linear SVMs model which partitions the feature space into several linearly inseparable subregions based on a probabilistic model, and the testing samples are classified by the weighted average over the mixture of classifiers. Li *et al.* [6] propose a piecewise linear classifier named "multiconlitron", which is based on the "convexly separable" concept. Qi *et al.* [7] partition the whole training samples into several local regions by applying the locality-sensitive hashing method, and construct the local classifier model in each region.

In this paper, we continue the fruitful line of the above thoughts and propose a novel classifier called H-MLSVMs. Specifically, it has a hierarchical tree-like structure with a mixture of linear SVMs at each node. The main advantages of the the proposed classifier are as follows. Firstly, H-MLSVMs does not require a large memory to store kernel values because H-MLSVMs is composed of LSVMs, and there are also very efficient algorithms for training LSVMs. Secondly, the hierarchical structure makes H-MLSVMs predict the labels of new arrived samples via a few of hyperplanes, and it is much faster than KSVMs. Thirdly, we quantify the generalization error bound for the class of LLSVMs based on the Rademacher complexity, and the stop criterion based on minimizing this bound ensures that H-MLSVMs can effectively avoid overfitting. Hence, H-MLSVMs has both the advantage of high speed comparable to LSVMs in the classification stage, and the advantage of good classification performance comparable to KSVMs.

2 Description of the H-MLSVMs

Before describing the proposed method, we first introduce the symbols used in the remaining parts of the paper.

- I: the node in the H-MLSVMs classifier. $I = [0]$ represents the root node; $[I, 1]$ and $[I, -1]$ are respectively the left and right child nodes of I.
- f_I: the classifier associated with the node I.
- S_I: subset of training samples in the node I.
- F_t: H-MLSVMs classifier in the t-th iteration.
- $V^{(t)}$: the evaluations of all training samples in the t-th iteration, and its i-th element is $V_i^{(t)} = F_t(\mathbf{x}_i)$.

2.1 Training Stage of H-MLSVMs

*This work is supported by NSFC (Grant Nos. 61100147, 61203241 and 61305035), Zhejiang Provincial Natural Science Foundation (Grants Nos. LY12F03016, LQ12F03004 and LQ13F030009).

†College of Mathematics and Information Sciences, Wenzhou University. Email: zhangxiaoqinnan@gmail.com.

2.1.1 Calculation of the training weights There are two major steps for calculating the training weights.

Step1: Training center selection Let the classifier evaluation of sample $\mathbf{x}_i$ be V_i in the current iteration, then the "centroid sample" is defined as:

$$\mathbf{x}_{\mathrm{c}} = \arg\max_{\mathbf{x}_i \in \hat{S}} \sum_{\mathbf{x}_j \in N_\varepsilon(\mathbf{x}_i)} \exp\left\{-\frac{\|\mathbf{x}_j - \mathbf{x}_i\|^2}{\sigma^2}\right\} E_{rr}^j \tag{2.1}$$

where $E_{rr}^j = \max\{0, \min\{1, 1 - y_j V_j / r\}\}$, $\hat{S} = \{x_i | y_i V_i < 1\}$, and $N_\varepsilon(\mathbf{x}_i)$ is the set of $\mathbf{x}_i$'s ε-neighborhoods with the same label as $\mathbf{x}_i$. From the definition of $\mathbf{x}_{\mathrm{c}}$ in (2.1), we can concluded that if a sample is selected as the "centroid sample", it must meet two conditions: (1) the sample density of its neighborhoods is high, (2) the samples in its neighborhoods have large "training errors". Therefore, the samples in local set $N_\varepsilon(\mathbf{x}_{\mathrm{c}})$ are currently misclassified "most".

Step2: Sample weights calculation The training weights of samples are calculated according to the sample similarities to the "centroid sample":

$$v_i = \exp\left\{-\frac{\|\mathbf{x}_i - \mathbf{x}_c\|^2}{\sigma^2}\right\} \Bigg/ \sum_{y_j = y_i} \exp\left\{-\frac{\|\mathbf{x}_j - \mathbf{x}_c\|^2}{\sigma^2}\right\} \tag{2.2}$$

Via (2.2), we assign the largest weights to the samples which are currently misclassified "most". This makes it more likely that the next classifier will correctly classify these samples.

2.1.2 Training Algorithm The training scheme of H-MLSVMs is summarized in Algorithm 1. In Step 7, we find the leaf node I which the "centroid sample" belongs to. This implies that the samples in set S_I are misclassified "most" by the current H-MLSVMs. Hence, we should assign a classifier to the node I to further split S_I. We define the classifier of node I by $f_I = \min\{f^{(t)}, f_{I_1}\}$ if $I_2 = 1$ (separate the negative samples from the classified "positive" samples), and by $f_I = \max\{f^{(t)}, f_{I_1}\}$ if $I_2 = -1$ (separate the positive samples from the classified "negative" samples), where I_1 is the parent node of I and I_2 is the last element of I. This setting method ensures the classifier associated node I considers not only the current trained LSVM $f^{(t)}$, but also the classifier at its parent node. Therefore, the classifier at each node is a mixture of maximums and minimums of LSVMs associated to its ancestor nodes. In Step 13, f_I splits S_I into subsets $S_{[I,1]}$ and $S_{[I,-1]}$ to enhance the impurity of current leaf nodes.

In the classification stage, we make the new arrived sample go through from the root node to a leaf node of H-MLSVMs. When the sample has arrived at a leaf node, if its evaluation is positive, we classify it as a positive sample, otherwise, we classify it as a negative sample.

REMARK 2.1. *The total number of classifiers at nodes of H-MLSVMs is a key parameter which determines the generalization performance for H-MLSVMs. We will give a proper stop criterion in Section 3.*

Algorithm 1 Training scheme of H-MLSVMs

Require: The training samples set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$.
Ensure: H-MLSVMs classifier F_t.

1: Train the standard LSVM classifier $f^{(1)}$ with sample set S, and assign $f^{(1)}$ to the root node, i.e., $f_{[0]} = f^{(1)}$. Initialize $t = 1$.
2: Let $F_1 = \{f_{[0]}\}$, and $\tilde{S}_1 = \{S_{[0,1]}, S_{[0,-1]}\}$, where $S_{[0,1]} = \{\mathbf{x} | f_{[0]}(\mathbf{x}) > 0, (\mathbf{x}, y) \in S\}$ and $S_{[0,-1]} = \{\mathbf{x} | f_{[0]}(\mathbf{x}) \leq 0, (\mathbf{x}, y) \in S\}$.
3: **loop**
4: Let t = t+1.
5: Find the "centroid sample" $\mathbf{x}_c$ via (2.1), and calculate training weights $\{v_i\}_{i=1}^n$ via (2.2).
6: Train the weighted LSVM $f^{(t)}$ with weighted samples set $S_v = \{(\mathbf{x}_i, y_i, v_i)\}_{i=1}^n$.
7: Find the leaf node which the "centroid sample" $\mathbf{x}_c$ belongs to, and denote by I. Let $I_2 = I(|I|)$ and $I_1 = I(1 : |I| - 1)$, where $|I|$ is the number of elements of I.
8: **if** $I_2 == -1$ **then**
9: Let $f_I = \max\{f^{(t)}, f_{I_1}\}$.
10: **else**
11: Let $f_I = \min\{f^{(t)}, f_{I_1}\}$.
12: **end if**
13: Let $F_t = F_{t-1} \cup \{f_I\}$, $S_{[I,1]} = \{\mathbf{x} | f_I(\mathbf{x}) > 0, \mathbf{x} \in S_I\}$, $S_{[I,-1]} = \{\mathbf{x} | f_I(\mathbf{x}) \leq 0, \mathbf{x} \in S_I\}$, $\tilde{S}_t = (\tilde{S}_{t-1} \cup \{S_{[I,1]}, S_{[I,-1]}\}) \setminus \{S_I\}$.
14: Update the evaluations of training samples $V^{(t)}$ by using current H-MLSVMs.
15: **end loop**

3 Generalization error bound

Because H-MLSVMs can be considered as an ensemble method of LLSVMs, we will quantify the generalization error bound for the class of LLSVMs based on the Rademacher complexity.

Assume a LLSVMs f defined on the domain $\mathcal{Z}$ is composed of κ LSVMs, denoted by

$$f = \{\langle \mathbf{w}_t, \mathbf{x} \rangle + b_t, \mathbf{x} \in \mathcal{Z}_t\}_{t=1}^{\kappa}. \tag{3.3}$$

$\mathcal{Z}_t$ is the subregion to which the LSVM $f_t(\mathbf{x}) = \langle \mathbf{w}_t, \mathbf{x} \rangle + b_t$ should be applied, such that $\mathcal{Z} = \cup_{t=1}^{\kappa} \mathcal{Z}_t$ and $\mathcal{Z}_i \cap \mathcal{Z}_j = \emptyset$ for $i \neq j$. Let $\mathfrak{F}_{B,\kappa,\mathcal{Z}_t}$ be the function class of LLSVMs defined by

$$\mathfrak{F}_{B,\kappa,\mathcal{Z}_t} = \left\{ f \,\middle|\, f = \{\langle \mathbf{w}_t, \mathbf{x} \rangle + b_t, \mathbf{x} \in \mathcal{Z}_t\}_{t=1}^{\kappa}, \sup_t \|\tilde{\mathbf{w}}_t\| \leq B \right\}$$

where $\tilde{\mathbf{w}}_t = [\mathbf{w}_t^\top \; b_t]^\top$. If we use the soft loss function

$$\ell_s(y, f(\mathbf{x})) = \min\left(1, \max\left(0, 1 - y f(\mathbf{x})/r\right)\right) \tag{3.4}$$

then the following theorem gives an generalization error bound for LLSVMs.

THEOREM 3.1. *Suppose training samples* $\mathbf{x}_{t1}, \cdots, \mathbf{x}_{tn_t}$ *are in the subregion* $\mathcal{Z}_t$ *for* $t = 1, \cdots, \kappa$. *Then for any fixed real value* $\delta \in (0, 1)$, *the expected risk of LLSVMs* $f \in \mathfrak{F}_{B,\kappa,\mathcal{Z}_t}$ *with probability at least* $1-\delta$ *can be bounded by*

$$L(f) \leq L_{emp}(f) + \frac{2B}{nr} \sum_{t=1}^{\kappa} \left(\sum_{i=1}^{n_t} \tilde{\mathbf{x}}_{ti}^{\top} \tilde{\mathbf{x}}_{ti} \right)^{\frac{1}{2}} + 3\sqrt{\frac{\ln(2/\delta)}{2n}}$$

where $L(f)$ *is the expected risk based on the* $0/1$ *loss,* $L_{emp}(f)$ *is the empirical risk based on the soft loss (3.4), and* $\tilde{\mathbf{x}}_{ti} = [\mathbf{x}_{ti}^{\top}\ 1]^{\top}$ *for* $i = 1, \cdots, n_t$ *and* $t = 1, \cdots, \kappa$.

Proof. Define the function class $\mathfrak{L}_{B,\kappa,\mathcal{Z}_t}$ by

$$\{\ell_s \,|\ell_s = \min(1, \max(0, 1 - yf(\mathbf{x})/r))\ , f \in \mathfrak{F}_{B,\kappa,\mathcal{Z}_t}\}\ .$$

Since $|\ell_s(f_1(\mathbf{x}), y) - \ell_s(f_2(\mathbf{x}), y)| \leq \frac{1}{r}|f_1(\mathbf{x}) - f_2(\mathbf{x})|$ for $\forall\ f_1, f_2 \in \mathfrak{F}_{B,\kappa,\mathcal{Z}_t}$, i.e., ℓ_s satisfies Lipschitz condition with constant $1/r$, then from Lemma 7 of [8], we have

$$\hat{R}_n(\mathfrak{L}_{B,\kappa,\mathcal{Z}_t}) \leq \hat{R}_n(\mathfrak{F}_{B,\kappa,\mathcal{Z}_t})/r \tag{3.5}$$

Let σ_{ti} be the Rademacher variable corresponding to the sample $\mathbf{x}_{ti}$, and let $\boldsymbol{\sigma}_t = [\sigma_{t1}, \ldots, \sigma_{tn_t}]$, then the empirical Rademacher complexity of $\mathfrak{F}_{B,\kappa,\mathcal{Z}_t}$ satisfies

$$\begin{aligned}
& \hat{R}_n(\mathfrak{F}_{B,\kappa,\mathcal{Z}_t}) \\
= \ & \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{f \in \mathfrak{F}_{B,\kappa,\mathcal{Z}_t}} \frac{1}{n} \sum_{t=1}^{\kappa} \sum_{i=1}^{n_t} \sigma_{ti} (\mathbf{w}_t^{\top} \mathbf{x}_{ti} + b_t) \right] \\
\leq \ & \frac{1}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{f \in \mathfrak{F}_{B,\kappa,\mathcal{Z}_t}} \sum_{t=1}^{\kappa} \left| \sum_{i=1}^{n_t} \langle \tilde{\mathbf{w}}_t, \sigma_{ti} \tilde{\mathbf{x}}_{ti} \rangle \right| \right] \\
\leq \ & \frac{B}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left[\sum_{t=1}^{\kappa} \left\| \sum_{i=1}^{n_t} \sigma_{ti} \tilde{\mathbf{x}}_{ti} \right\| \right] \\
\leq \ & \frac{B}{n} \sum_{t=1}^{\kappa} \left[\mathbb{E}_{\boldsymbol{\sigma}_t} \left(\sum_{i,j=1}^{n_t} \sigma_{ti} \sigma_{tj} \tilde{\mathbf{x}}_{ti}^{\top} \tilde{\mathbf{x}}_{tj} \right) \right]^{\frac{1}{2}} \\
= \ & \frac{B}{n} \sum_{t=1}^{\kappa} \left(\sum_{i=1}^{n_t} \tilde{\mathbf{x}}_{ti}^{\top} \tilde{\mathbf{x}}_{ti} \right)^{\frac{1}{2}}
\end{aligned}$$

The disappearance of the mixed terms $\sigma_{ti}\sigma_{tj}\tilde{\mathbf{x}}_{ti}^{\top}\tilde{\mathbf{x}}_{tj}$ for $i \neq j$ in the last equation follows from the fact that the four possible combinations of -1 and 1 have equal probability with two of the four having the opposite signs and hence cancelling out. On the other hand,

$$L(f) = \mathbb{E}_D[\ell_{0/1}(f(\mathbf{x}), y)] \leq \mathbb{E}_D[\ell_s(f(\mathbf{x}), y)]$$

Then according to the Theorem 4.9 in [9], the proof is complete.

Theorem 3.1 gives us an upper bound on the generalization error in terms of the empirical risk and the complexity of the function class $\mathfrak{F}_{B,\kappa,\mathcal{Z}_t}$, where we measure the complexity of $\mathfrak{F}_{B,\kappa,\mathcal{Z}_t}$ by the minimal "margin" of all LSVMs ($2/\|\tilde{\mathbf{w}}_t\|$ is the margin of the t-th LSVM in the extended feature space $\tilde{\mathcal{X}} = \{\tilde{\mathbf{x}} | \tilde{\mathbf{x}} = [\mathbf{x}^{\top} 1]^{\top}\}$) and the partition of the training space. From the upper bound, we use the minimization of

$$\begin{aligned}
& \sum_{t=1}^{\kappa} \sum_{i=1}^{n_t} \min(1, \max(0, 1 - y_{ti} f(\mathbf{x}_{ti})/r)) \\
& + \lambda \left(\sup_t \|\tilde{\mathbf{w}}_t\| \right) \cdot \sum_{t=1}^{\kappa} \left[\mathrm{tr} \left(Z_t^{\top} \tilde{X}^{\top} \tilde{X} Z_t \right) \right]^{\frac{1}{2}}
\end{aligned} \tag{3.6}$$

as the stop criterion, where λ is a tunable parameter and the second term in (3.6) can be seen as a regularization for the complexity of $\mathfrak{F}_{B,\kappa,\mathcal{Z}_t}$. In the algorithm of H-MLSVMs, we train a series of LSVMs with the total number of classifiers at nodes of H-MLSVMs in the increasing order, and the repetition is not stopped until the minimization of the value in (3.6) is achieved. The regularization term ensures that overfitting can be effectively avoided.

4 Experiments

In this section, H-MLSVMs is compared with: (1) LSVM, (2) adaboost with decision stump as weak learners (AdaStu), (3) mixing linear SVMs (MLSVM) [5], (4) LIBSVM (with RBF kernel $k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2}$). The parameters are obtained by using 5-fold cross validation. A list of the datasets and the optimal parameters is given in Table 1. C_{LIB} and γ_{LIB} are parameters for LIBSVM. γ_{ML} is the scale parameter for MLSVM. σ^2 and λ are parameters for H-MLSVMs.

We conducted the experiments on two synthetic and five real-world datasets that are publicly available. Fig. 1 shows that H-MLSVMs achieves good performance comparable to KSVMs on the two highly nonlinear datasets. Table 2 gives the test accuracy of the five methods, and Table 3 gives the mean number of support vectors used by LIBSVM, and mean number of linear classifiers employed by AdaStu and MLSVM, together with the mean number of H-MLSVMs encountered per testing sample [1]. From Table 2, it can be seen clearly that LIBSVM outperform all other methods, however, they are at the cost of generating the largest number of SVs (as shown in Table 3), and this makes them have the highest computational complexity in classification stage. MLSVM reports considerable reductions of the number of linear classifiers, but they are

[1] The codes of LIBSVM are implemented in C++ and the codes of Ada-Stump, MLSVM, and H-MLSVMs are implemented in MATLAB, hence we prefer the number of SVs or linear classifiers to the classification time in fairness. Note that the complexity of MLSVM scales with twice the number of linear classifiers, including gate function and classifier evaluations.

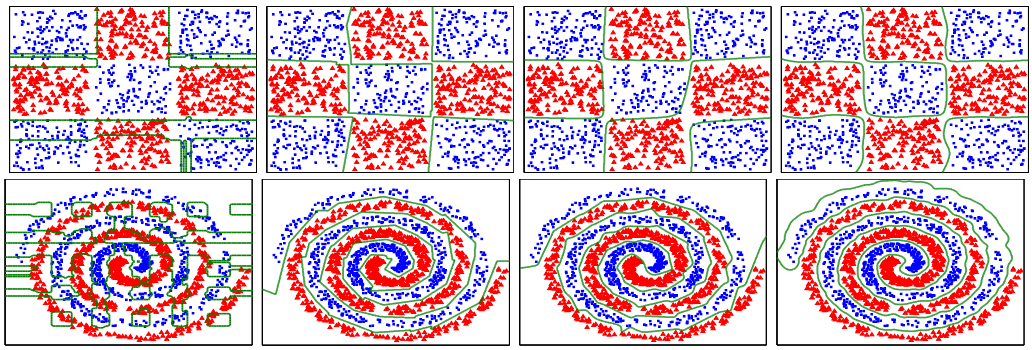


Figure 1: Decision boundaries of AdaStu, H-MLSVMs, MLSVM and LIBSVM on the two synthetic datasets.

Table 1: Overall description of data sets.

Data	Dim.	TR#	C_{LIB}	γ_{LIB}	γ_{ML}	σ^2	λ
Check	2	1500	10	1	5	0.1	0.01
Spiral	2	2000	10	1	0.5	0.5	0.001
Monks	6	556	10	1	0.05	0.01	0.1
Letter	16	1536	10	1	0.1	0.6	0.001
Digit	16	2198	10	1	0.1	0.6	0.05
Chess	6	6179	10	0.1	1	4	0.001
Stat	36	1410	10	0.1	10	1	0.001

Table 2: Testing accuracy of the five method.

Data	LSVM	LIBSVM	AdaStu	MLSVM	Ours
Check	52.8(2.3)	99.6(0.3)	74.8(2.4)	98.6(1.3)	99.1(0.6)
Spiral	55.1(2.1)	100(0)	70.4(1.7)	99.4(0.2)	99.7(0.3)
Monks	66.3(3.7)	99.5(0.8)	75.1(3.4)	97.5(1.9)	99.8(0.5)
Letter	97.7(0.6)	99.9(0.1)	99.4(0.3)	99.2(0.7)	99.7(0.2)
Digit	97.1(1.0)	99.8(0.2)	97.5(0.9)	99.6(0.1)	99.4(0.3)
Chess	75.6(1.1)	96.6(0.3)	84.2(1.8)	92.3(3.9)	92.6(1.1)
Stat	95.8(1.2)	98.3(0.7)	93.6(1.9)	96.5(2.6)	97.2(1.2)

still much higher than that of H-MLSVMs. H-MLSVMs achieves testing accuracy closer to the LIBSVM than that of the other methods (except for MLSVM on data `Digit`). More importantly, H-MLSVMs classifies a testing sample only via several (at most 8.3) LSVMs (as shown in Table 3), so the time cost of H-MLSVMs in the classification stage is significantly reduced. This is the main focus of H-MLSVMs, which is not having the best testing accuracy but providing a method capable of classifying a pattern in a few milliseconds while obtaining a competitive performance. Thanks to the regularization term in (3.6), this ensures that the H-MLSVMs do not fall into overfitting and have the good generalization ability comparable to KSVMs. The hierarchical structure of H-MLSVMs makes it achieve the high efficiency in the classification stage.

5 Conclusions

In this paper, we proposed a novel classifier called H-MLSVMs. The hierarchical structure of H-MLSVMs and the regularization term in (3.6) make it have the high speed comparable to LSVM in the classification stage and the good classification performance comparable to KSVMs. Experiments on both synthetic and real datasets are presented to show the effectiveness of our method. In the future, we will focus on the online learning of H-MLSVMs.

Table 3: Testing complexity in the classification stage.

Data	LIBSVM	AdaStu	MLSVM	Ours
Check	160.7(4.6)	116.4(61.7)	10(0)	6.8(0.7)
Spiral	616.0(10.3)	143.1(29.7)	50(0)	8.3(0.3)
Monks	290.2(5.3)	40.8(20.0)	20(0)	4.1(0.4)
Letter	98.5(5.6)	79.5(47.1)	20(0)	4.3(1.5)
digit	89.3(4.1)	69.9(60.9)	20(0)	3.8(0.4)
Chess	879.9(11.0)	174.7(25.5)	50(0)	8.2(0.7)
Stat	108.1(3.9)	52.3(48.9)	40(0)	4.9(1.2)

References

[1] A. Rodriguez, V. N. Boddeti, B. V. Kumar, and A. Mahalanobis, *Maximum margin correlation filter: A new approach for localization and classification*, IEEE Trans. Image Process., 22 (2013), pp. 631–643.

[2] Y. Li, L. Du, and H. Liu, *Hierarchical classification of moving vehicles based on empirical mode decomposition of micro-doppler signatures*, IEEE Trans. Geosci Remote., 51 (2013), pp. 3001–3013.

[3] Y. Freund and R. E. Schapire, *A decision-theoretic generalization of on-line learning and an application to boosting*, J. Comput. Syst. Sci., 55 (1997), pp. 119–139.

[4] L. Breiman, *Random forests*, Mach. Learn., 45 (2001), pp. 5–32.

[5] Z. Y. Fu, A. Robles-Kelly, and J. Zhou, *Mixing linear SVMs for nonlinear classification*, IEEE Trans. Neural Networ., 12 (2010), pp. 1963–1975.

[6] Y. Li, B. Liu, X. Yang, Y. Fu, and H. Li, *Multiconlitron: a general piecewise linear classifier*, IEEE Trans. Neural Networ., 2 (2011), pp. 276–289.

[7] G. Qi, T. Qi, and T. Huang, *Locality-sensitive support vector machine by exploring local correlation and global regularization*, CVPR, IEEE Press, 2011.

[8] R. Meir and T. Zhang, *Generalization error bounds for bayesian mixture algorithms*, J. Mach. Learn. Res., 4 (2003), pp. 839–860.

[9] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University, New York, 2004.

Adaptive Clustering for Monitoring Distributed Data Streams

Maria Barouti* Daniel Keren† Jacob Kogan‡ Yaakov Malinovsky§

Abstract

Monitoring data streams in a distributed system is a challenging problem with profound applications. The task of feature selection (e.g., by monitoring the information gain of various features) is an example of an application that requires special techniques to avoid a very high communication overhead when performed using straightforward centralized algorithms. The proposed approach enables monitoring values of a threshold function over distributed data streams through a set of constraints applied independently on each cluster of streams. The clusters are designed to adapt themselves to the data stream. We report experiments with clustering approach that yield communication load reduction.

Keyword list: adaptive stream mining, convex analysis, distributed system, clustering

1 Introduction

In many emerging applications one needs to process a continuous stream of data in real time. The current contribution is motivated by results reported in [4], where a more general type of monitoring query is described as follows:

Let $\mathbf{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_n\}$ be a set of data streams collected at n nodes $\mathbf{N} = \{\mathbf{n}_1, \ldots, \mathbf{n}_n\}$. Let $\mathbf{v}_1(t), \ldots, \mathbf{v}_n(t)$ be d-dimensional, real-valued, time varying vectors derived from the streams. For a function $f : \mathbf{R}^d \to \mathbf{R}$ we would like to monitor the inequality

$$f\left(\frac{\mathbf{v}_1(t) + \ldots + \mathbf{v}_n(t)}{n}\right) > 0 \tag{1.1}$$

while minimizing communication between the nodes. In e.g. [4, 3] a few real-life applications of this monitoring problem are described; see also Section 2 here.

The present paper deals with the information gain function (see Section 2 for details). Rather than focus on the values of f we consider the location of the vectors $\mathbf{v}_i(t)$ relative to the set $\mathbf{Z}_+(f) = \{\mathbf{v} : f(\mathbf{v}) > 0\}$. We restate (1.1) as

$$\mathbf{v}(t) = \frac{\mathbf{v}_1(t) + \ldots + \mathbf{v}_n(t)}{n} \in \mathbf{Z}_+(f). \tag{1.2}$$

As a simple illustration, consider the case of three scalar functions $v_1(t)$, $v_2(t)$ and $v_3(t)$, and the identity function f (i.e. $f(x) = x$). We would like to monitor the inequality

$$v(t) = \frac{v_1(t) + v_2(t) + v_3(t)}{3} > 0$$

while keeping the nodes silent as long as possible. One strategy is to verify the initial inequality $v(t_0) = \frac{v_1(t_0) + v_2(t_0) + v_3(t_0)}{3} > 0$ and to keep the nodes silent while

$$|v_i(t) - v_i(t_0)| < \delta = v(t_0),\ t \geq t_0,\ i = 1, 2, 3.$$

The first time t when one of the functions, say $v_1(t)$, crosses the boundary of the local constraint, i.e. $|v_1(t) - v_1(t_0)| \geq \delta$ the nodes communicate, t_1 is set to be t, the mean $v(t_1)$ is computed, the local constraint δ is updated and made available to the nodes. The nodes are kept silent as long as the inequalities

$$|v_i(t) - v_i(t_1)| < \delta,\ t \geq t_1,\ i = 1, 2, 3$$

hold [5]. The numerical experiments conducted in [5] with the dataset described in Section 5 show that (a) the number of time instances the mean violates (1.1) is a small fraction ($< 1\%$) of the number of time instances when the local constraint is violated at the nodes, (b) the lion's share of communications (about 75%) is required because of a single node violation of the local constraint δ.

If, for example, the local constraint is violated at $\mathbf{n}_1$, i.e. $|v_1(t) - v_1(t_0)| \geq \delta$, and at the same time

$$v_1(t) - v_1(t_0) = -[v_2(t) - v_2(t_0)], \text{while } |v_3(t) - v_3(t_0)| < \delta$$

then $|v(t) - v(t_0)| < \delta$, $f(v(t)) > 0$, and update of the mean can be avoided. Separate monitoring of the two

*Math. and Stat., UMBC, Baltimore, MD 21250, bmaria2@umbc.edu

†Department of Computer Science, Haifa University, Haifa 31905, Israel, dkeren@cs.haifa.ac.il

‡Math. and Stat., UMBC, Baltimore, MD 21250, kogan@umbc.edu

§Math. and Stat., UMBC, Baltimore, MD 21250, yaakovm@umbc.edu

node cluster $\{\mathbf{n}_1, \mathbf{n}_2\}$ would require communication involving two nodes only, and could reduce communication load. In this paper we advance clustering approach to monitoring. A specific clustering strategy applicable with a variety of norms leading to communication savings is the main contribution of this work.

In Section 2 we present a relevant Text Mining application. Section 3 provides motivation for node clustering. A specific implementation of node clustering is presented in Section 4. Experimental results are reported in Section 5.

2 Text Mining application

Let $\mathbf{T}$ be a textual database (for example a collection of mail or news items). We denote the size of the set $\mathbf{T}$ by $|\mathbf{T}|$. We will be concerned with two subsets of $\mathbf{T}$:

1. $\mathbf{R}$–the set of "relevant" texts (e.g. texts not labeled as "spam"),
2. $\mathbf{F}$–the set of texts that contain a "feature" (word or term for example).

We denote complements of the sets by $\overline{\mathbf{R}}$, $\overline{\mathbf{F}}$ respectively (i.e. $\mathbf{R} \cup \overline{\mathbf{R}} = \mathbf{F} \cup \overline{\mathbf{F}} = \mathbf{T}$), and consider the relative size of the four sets $\mathbf{F} \cap \overline{\mathbf{R}}$, $\mathbf{F} \cap \mathbf{R}$, $\overline{\mathbf{F}} \cap \overline{\mathbf{R}}$, and $\overline{\mathbf{F}} \cap \mathbf{R}$ as follows:

$$x_{11}(\mathbf{T}) = \frac{|\mathbf{F} \cap \overline{\mathbf{R}}|}{|\mathbf{T}|}, \quad x_{12}(\mathbf{T}) = \frac{|\mathbf{F} \cap \mathbf{R}|}{|\mathbf{T}|}, \tag{2.3}$$
$$x_{21}(\mathbf{T}) = \frac{|\overline{\mathbf{F}} \cap \overline{\mathbf{R}}|}{|\mathbf{T}|}, \quad x_{22}(\mathbf{T}) = \frac{|\overline{\mathbf{F}} \cap \mathbf{R}|}{|\mathbf{T}|}.$$

Note that $0 \le x_{ij} \le 1$, and $x_{11} + x_{12} + x_{21} + x_{22} = 1$. The function f is given by

$$\sum_{i,j} x_{ij} \log \left(\frac{x_{ij}}{(x_{i1} + x_{i2})(x_{1j} + x_{2j})} \right), \tag{2.4}$$

where $\log x = \log_2 x$ throughout the paper. The *information gain* for the "feature" is provided by f [1].

As an example, we consider n agents installed on n different servers, and a stream of texts arriving at the servers. Let $\mathbf{T}_h = \{\mathbf{t}_{h1}, \ldots, \mathbf{t}_{hw}\}$ be the last w texts received at the h^{th} server, with $\mathbf{T} = \bigcup_{h=1}^{n} \mathbf{T}_h$. Note that

$$x_{ij}(\mathbf{T}) = \sum_{h=1}^{n} \frac{|\mathbf{T}_h|}{|\mathbf{T}|} x_{ij}(\mathbf{T}_h),$$

i.e., entries of the global contingency table $\{x_{ij}(\mathbf{T})\}$ are the weighted average of the local contingency tables $\{x_{ij}(\mathbf{T}_h)\}$, $h = 1, \ldots, n$.

3034	620	162	70	38	26	34	17	5	0
1	2	3	4	5	6	7	8	9	10

Table 1: first row–number of time instances when a local constraint has been violated by exactly k nodes, $k = 1, 2, \ldots, 10$; second row–number of nodes–violators, $r = 0.0025$, l_2 norm, the feature is "bosnia"

To check that the given "feature" is sufficiently informative with respect to the target relevance label r, one may want to monitor the inequality

$$f\left(x_{11}(\mathbf{T}), x_{12}(\mathbf{T}), x_{21}(\mathbf{T}), x_{22}(\mathbf{T})\right) - r > 0 \tag{2.5}$$

while minimizing communication between the servers.

3 Clustering for monitoring: motivation

In what follows we denote a norm of a vector $\mathbf{v}$ by $\|\mathbf{v}\|$. A specific choice of a norm will be indicated when needed.

The monitoring strategy proposed in [5] and applied to data streams generated from the data described in Section 5 leads to 4006 time instances in which the local constraints are violated, and the root is updated. Table 1 shows that in 3034 out of 4006 time instances, communications with the root are triggered by constraint violations at exactly one node.

The results suggest to cluster nodes to reduce communication load. Each cluster π will be equipped with a "coordinator" $\mathbf{c}$. If $\mathbf{n}' \in \pi$ violates its local constraint at time t, then the coordinator collects vectors $\mathbf{v_n}(t) - \mathbf{v_n}(t_i)$ from all $\mathbf{n} \in \pi$, computes the mean, and checks whether the mean violates the coordinator constraint $\delta(\mathbf{c})$. We shall follow [4] and refer to this step as "the balancing process." If $\delta(\mathbf{c})$ is violated, the mean of the entire dataset is recomputed.

A standard clustering problem is often described as "... finding and describing cohesive or homogeneous chunks in data, the clusters" [2]. For the problem at hand we would like to partition the set of nodes $\mathbf{N}$ into k disjoint clusters $\Pi = \{\pi_1, \ldots, \pi_k\}$. If $\frac{1}{|\pi_i|} \left\| \sum_{\mathbf{n} \in \pi_i} [\mathbf{v_n}(t) - \mathbf{v_n}(t_j)] \right\| < \delta$ for each i, then one has

$$\left\| \sum_{\mathbf{n} \in \mathbf{N}} \frac{\mathbf{v_n}(t) - \mathbf{v_n}(t_j)}{n} \right\| \le \sum_{i=1}^{k} \frac{|\pi_i|}{n} \left\| \sum_{\mathbf{n} \in \pi_i} \frac{\mathbf{v_n}(t) - \mathbf{v_n}(t_j)}{|\pi_i|} \right\| < \delta.$$

Hence the "new" mean $\frac{1}{n} \sum_{\mathbf{n} \in \mathbf{N}} \mathbf{v_n}(t)$ belongs to $\mathbf{Z}_+(f)$ if the "old" mean $\frac{1}{n} \sum_{\mathbf{n} \in \mathbf{N}} \mathbf{v_n}(t_j)$ belongs to this set. A

possible definition for quality of a partition Π is

$$(3.6)\ Q(\Pi)=\max_{i\in\{1,\ldots,k\}}\left\{\frac{1}{|\pi_i|}\left|\left|\sum_{\mathbf{n}\in\pi_i}[\mathbf{v_n}(t)-\mathbf{v_n}(t_j)]\right|\right|\right\}$$

Our aim is to identify k and a k cluster partition Π^o that **minimizes** (3.6). Our monitoring problem requires to assign nodes $\{\mathbf{n}_{i_1},\ldots,\mathbf{n}_{i_l}\}$ to the same cluster π so that the total average change within cluster

$$\left|\left|\frac{1}{|\pi|}\sum_{\mathbf{n}\in\pi}[\mathbf{v_n}(t)-\mathbf{v_n}(t_j)]\right|\right|\ \text{ for } t>t_j$$

is minimized. Hence, unlike classical clustering procedures, this one needs to combine "dissimilar" nodes together.

The proposed partition quality $Q(\Pi)$ (see (3.6)) generates three immediate problems:

1. The single cluster partition always minimizes $Q(\Pi)$.
2. Computation of $Q(\Pi)$ involves future values $\mathbf{v_n}(t)$, which are not available at time t_j when the clustering is performed.
3. The individual clusters' sizes should affect the clustering quality to account for the balancing process communication.

4 Clustering for monitoring: implementation

We define the quality of the cluster π by

$$(4.7)\quad q(\pi)=\frac{1}{|\pi|}\left|\left|\sum_{\mathbf{n}\in\pi}[\mathbf{v_n}(t)-\mathbf{v_n}(t_j)]\right|\right|+\alpha|\pi|,$$

where $\alpha\geq 0$ is a scalar parameter. The quality of the partition Π is defined by

$$(4.8)\qquad Q(\Pi)=\max\{q(\pi_i)\ :\ i=1,\ldots,k\}.$$

When $\alpha=0$ the partition that minimizes $Q(\Pi)$ is a single cluster partition (that we would like to avoid). When $\max_{\mathbf{n}\in\mathbf{N}}||\mathbf{v_n}(t)-\mathbf{v_n}(t_j)||=m\leq\alpha$ the optimal partition is made up of n singleton clusters. We focus on $0<\alpha<m$ that depends on t and t_j, and show below how to avoid this dependence.

In order to compute $Q(\Pi)$ at time t_j one needs to know $\mathbf{v_n}(t)$ at a future time $t>t_j$ which is not available. We shall use past values of $\mathbf{v_n}(t)$ for prediction. For each node $\mathbf{n}$ we build "history" vectors $\mathbf{h_n}(t_j)$ that accumulate the weighted history of changes, with the current change carrying a weight twice as much as the previous one. When the vector set is normalized by the magnitude of the longest vector in the set, the range for α conveniently shrinks to $[0,1]$, and the induced optimal partitioning remains the same. In what follows we set $h=\max_{\mathbf{n}\in\mathbf{N}}||\mathbf{h_n}(t_j)||$, assume that $h>0$, and describe a clustering procedure for the normalized vector set

$$\{\mathbf{a}_1,\ldots,\mathbf{a}_n\},\ \mathbf{a}_i=\frac{1}{h}\mathbf{h}_{\mathbf{n}_i}(t_j),\ i=1,\ldots,n.$$

We start with the n cluster partition Π^n (each cluster is a singleton). If a k cluster partition Π^k, $k>2$ is already available we:

1. Identify the partition cluster π_j with maximal norm of its vectors' mean.
2. Identify cluster π_i so that the merger of π_i with π_j produces a cluster of smallest possible quality (4.7).

The partition Π^{k-1} is obtained from Π^k by merging clusters π_j and π_i. The final partition is selected from the $n-1$ partitions $\{\Pi^2,\ldots,\Pi^n\}$ as the one that minimizes Q given by (4.8).

If there is reason to believe that e.g. the following inequality

$$(4.9)\qquad 2\|\mathbf{v}_1(t)-\mathbf{v}_1(t_i)\|\leq\|\mathbf{v}_2(t)-\mathbf{v}_2(t_i)\|$$

always holds, then the number of node violations may be reduced by imposing node dependent constraints

$$\|\mathbf{v}_1(t)-\mathbf{v}_1(t_i)\|<\delta_1=\frac{2}{3}\delta,\ \text{and}\ \|\mathbf{v}_2(t)-\mathbf{v}_2(t_i)\|<\delta_2=\frac{4}{3}\delta$$

so that the wider varying stream at the second node enjoys larger "freedom" of change, while the inequality

$$\left|\left|\frac{\mathbf{v}_1(t)+\mathbf{v}_2(t)}{2}-\frac{\mathbf{v}_1(t_i)+\mathbf{v}_2(t_i)}{2}\right|\right|<\frac{\delta_1+\delta_2}{2}=\delta$$

holds true. Assigning "weighted" local constraints requires information provided by (4.9). With no additional assumptions about the stream data distribution this information is not available. We estimate the weights through past values $\|\mathbf{v}_j(t)-\mathbf{v}_j(t_i)\|$.

1. Start with the initial set of weights

$$(4.10)\quad w_1=\ldots=w_n=1,\ W_1=\ldots=W_n=1.$$

2. At the next time t, each node $\mathbf{n}_j$ computes updates

$$W_j=\frac{1}{2}W_j+\|\mathbf{v}_j(t)-\mathbf{v}_j(t_i)\|,\ \text{with}\ W_j(t_0)=1.$$

Next time when the distance δ from the mean to the boundary of $\mathbf{Z}_+(f)$ is updated, each node $\mathbf{n}_j$ broadcasts W_j to the root, the root computes $W=\sum_{j=1}^{n}W_j$, and transmits the updated $\delta(\mathbf{n}_j)=w_j\delta$ where $w_j=n\times\frac{W_j}{W}$ back to node j. For a coordinator $\mathbf{c}$ of a node cluster π the constraint $\delta(\mathbf{c})=\frac{1}{|\pi|}\sum_{\mathbf{n}\in\pi}\delta(\mathbf{n})$.

norm	mean updates	broadcasts
l_1	2591	67388
l_2	3140	81650
l_∞	3044	79144

Table 2: Number of mean computations, and broadcasts for feature "febru" with threshold $r = 0.0025$, no clustering

norm	alpha	root mean update	coordinator mean update	total broadcasts
l_1	0.70	1431	0	38665
l_2	0.80	1317	0	35597
l_∞	0.65	1409	0	38093

Table 3: Number of root and coordinator mean computations, and total broadcasts for feature "febru" with threshold $r = 0.0025$ with clustering

norm	mean updates	broadcasts
l_1	3053	79378
l_2	4006	104156
l_∞	3801	98826

Table 4: Number of mean computations, and broadcasts, for feature "bosnia" with threshold $r = 0.0025$, no clustering

norm	alpha	root mean update	coordinator mean update	total broadcasts
l_1	0.65	3290	2	89128
l_2	0.55	3502	7	97602
l_∞	0.60	3338	2	91306

Table 5: Number of root and coordinator mean computations, and total broadcasts for feature "bosnia" with threshold $r = 0.0025$ and clustering

5 Experimental results

The data streams analyzed in this section are generated from the Reuters Corpus RCV1–V2. The data is available from `http://leon.bottou.org/projects/sgd` and consists of $781,265$ tokenized documents.

Each document is labeled as belonging to one or more categories. We follow [4] and label a vector as "relevant" if it belongs to the "CORPORATE/INDUSTRIAL" ("CCAT") category, and "spam" otherwise. Following [4] we focus on three features: "bosnia," "ipo," and "febru." Each experiment was performed with 10 nodes, where each node holds a sliding window containing the last 6,700 documents it received.

First we use $67,000$ documents to generate initial sliding windows. The remaining $714,265$ documents are used to generate datastreams, hence the selected feature information gain is computed $714,265$ times. For the experiments described below, the threshold value r is predefined, and the goal is to monitor the inequality $f(\mathbf{v}) - r > 0$ while minimizing communication between the nodes.

The numerical experiment reported in [5] with the feature "febru," and the threshold $r = 0.0025$ are shown in Table 2 where a broadcast is defined as one time transmission of information between different nodes. We run the node clustering monitoring for the same feature and threshold with $\alpha = 0.05, 0.10, \ldots, 0.95$. The best result for l_1, l_2, and l_∞ norms with respect to α are presented in Table 3, and the result shows about 50% decrease in the number of broadcasts.

Next we turn to the features "ipo" and "bosnia." While communication savings for "ipo" are similar to those presented above for "febru", an application of clustering to monitoring "bosnia's" information gain appears to be far less successful. Results obtained without clustering in [5] are presented in Table 4. Application of the clustering procedure leads to a slight decrease in the number of broadcasts in case of the l_2 and l_∞ norms (see Table 5). In case of the l_1 norm, the number of broadcasts increases. This is just a reminder that clustering is no universal remedy, and in some cases better performance is achieved with no clustering.

References

[1] Gray, R.M.: Entropy and Information Theory. Springer–Verlag, New York, (1990)

[2] Mirkin, B.: Clustering for Data Mining: A Data Recovery Approach. Chapman & Hall/CRC, Boca Raton, (2005)

[3] Gabel, M. and Schuster, A. and Keren, D.: Communication-efficient outlier detection for scale-out systems. In BD3@VLDB, 19-24, (2013)

[4] I. Sharfman and A. Schuster and D. Keren: A Geometric Approach to Monitoring Threshold Functions over Distributed Data Streams, ACM Transactions on Database Systems, **32**, 23:1-23:29, (2007)

[5] Kogan, J.: Feature Selection over Distributed Data Streams through Convex Optimization. Proceedings of the Twelfth SIAM International Conference on Data Mining (SDM 2012), SIAM, 475-484, (2012)

Exploratory Data Analysis for Frequent Sequence Mining

Adam Perer*

Abstract

Extracting insights from temporal event sequences is an important challenge for many domains. However, many current techniques for mining frequent patterns are ineffective for exploratory data analysis, as the mining algorithms are often not integrated with an interactive interface. To address this issue, we designed Frequence, a visual analytics system that integrates data mining and visualization for finding frequent patterns from longitudinal event sequences. Frequence features a novel frequent sequence mining algorithm, as well as a visual interface designed to support insights. To support exploration, users can adapt and modify the mining algorithm according to their needs, and mine patterns on the level-of-detail relevant to their current task. More details about Frequence's visualization and the novel mining algorithm are described in [1].

Frequence supports finding useful patterns from a variety of domains, such as frequent sequences from Foursquare to understand patterns of urban mobility (Figure 1) and event sequences from clinical records to understand the progression of diseases (Figure 2). In this workshop, I hope to demonstate the exploratory power of integrating data mining and visualization and showcase how such an integration can lead to novel insights.

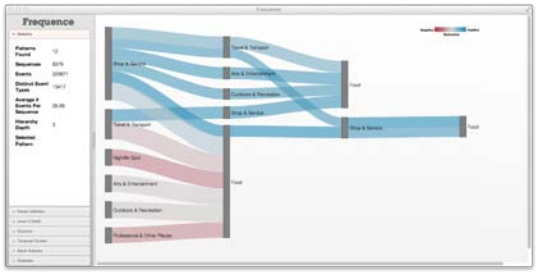


Figure 1: The Frequence UI visualizing patterns mined from a Foursquare dataset at a coarse level-of-detail. Events in the frequent sequences are represented as nodes, and event nodes that belong to the same sequence are connected by edges. The visualization uses color to encode each patterns association with an associated outcome. Here, the patterns that occur more often with popular users (those who have a lot of followers on Twitter) are more blue. The patterns that occur more often with unpopular users (those who have few followers) are more red. The neutral patterns that appear common to both popular and unpopular users are gray.

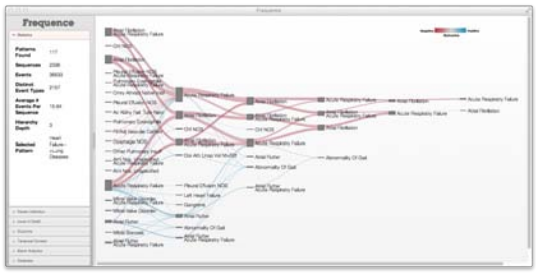


Figure 2: The Frequence UI visualizing patterns mined from a lung disease patient dataset with Frequence at a fine level-of-detail. Here, blue patterns occur among healthy patients, whereas red patterns occur more often in patients with a negative outcome. Interestingly, the patterns clearly diverge into red and blue, making it clear that what happens immediately after certain events will likely determine if the patient's health with improve or not.

References

[1] A. Perer and F. Wang, *Frequence: Interactive Mining and Visualization of Temporal Frequent Event Sequences*, ACM International Conference on Intelligent User Interfaces (IUI), (2014).

*IBM T.J. Watson Research Center

Case Study: Model-based vs Distance-based Search in Time Series Databases

Alexios Kotsifakos
Department of Computer Science and Engineering
University of Texas at Arlington, USA
alexios.kotsifakos@mavs.uta.edu

Abstract

Time series data are daily produced at an unprecedented rate in almost every application domain, such as stock markets, sensor networks, astronomy, and medicine. Hence, there has been a significant amount of research on mining and querying time series, including classification, clustering, approximation, and indexing among others. A topic of particular interest is to search large time series databases to identify content of interest. In this work, we explore two approaches for such searches: a distance-based approach, where what we would like to identify is described with a single time series, and a model-based approach, where the query is a model, which represents a class of time series and in our case is a Hidden Markov Model (HMM). We first evaluated the NN classification error rate of HMMs compared with four widely known distance measures, DTW, cDTW, ERP, and MSM on the training sets of all 45 time series datasets of the UCR archive. Modeling time series with HMMs achieves lower error rates than the competitors in 28 datasets and equal in 4. Secondly, we evaluated the five search methods on the test sets of the 45 datasets, and observed that HMMs provide more accurate results if there is a sufficient number of time series to train them. With regard to efficiency, cDTW is the fastest method among the competitors, but its accuracy is much worse than the other methods.

1 Introduction

Time series data have become ubiquitous during the last decades. Sequences of numerical measurements are produced at regular or irregular time intervals in vast amounts in almost every application domain, such as stock markets, medicine, sensor networks, moving objects, scientific experiments, and biology. As a result, effective and efficient querying and mining of time series is extremely important and there is still a need for new techniques for classification, clustering, indexing, and approximation of time series [6].

Large databases of time series can be exploited so as to extract knowledge on what has happened in the past or to recognize what is happening in the present. Assume that we want to *search* the database for time series corresponding to a specific activity, such as "person falling down". The search process must decide, for each time series, whether that time series is a good match for this particular search (of instances of "person falling down") or not. This can be done as follows: (a) assign a score to each time series in the database, which indicates how good a match each time series is for the particular search we are conducting, (b) rank database time series according to their score, (c) return to the user the top K matches, where K is a user-specified parameter.

The most critical step of the aforementioned procedure is the first one. Any mathematically valid scoring function can be used, but the choice will essentially determine the accuracy of the results. Thus, it is of particular importance to assign the best scores to the time series that the user would consider to be "correct matches" for the query.

In distance-based search, to perform our search we provide as a query a time series, e.g., time series corresponding to the "falling down" activity. Then, the score assigned to each database time series is the distance/similarity score that is computed using the selected measure. In model-based search, we specify what we are looking for by submitting as a query a model of the activity. This model can be a *Hidden Markov Model* (*HMM*), which has been trained on time series, e.g., of the "falling down" activity. The score that is assigned to each database time series is the output of the model on that time series. For HMMs, this score can be computed through the Forward algorithm [19] taking as input the HMM and the time series.

The **main contributions** of this paper include: (a) A way of representing time series of a specific class via an HMM, and a comparative evaluation of this repre-

sentation against four distance measures, DTW, cDTW, ERP, and MSM in terms of classification accuracy on the training sets of 45 datasets [8]. The evaluation shows that HMMs can attain significantly higher accuracy in 17 datasets, relatively higher accuracy in 11, and equal accuracy in 4; hence better or equal accuracy in 32 datasets. (b) An extensive experimental evaluation of model-based search with HMMs and distance-based search with DTW, cDTW, ERP, and MSM, on 45 datasets in terms of precision vs. recall and runtime. We observed that HMMs can produce significantly better tradeoffs than the competitors when trained with a sufficient number of training time series, while they are usually slower than the distance-based methods.

2 Related Work

We provide an overview on distance measures for whole sequence matching and time series representations.

2.1 Time Series Distance Measures Several distance/similarity methods for time series have been proposed in the literature [24]. Dynamic Time Warping (DTW) [12] is not only a widely used distance measure for computing distances between time series, but also provides very good classification accuracy results. Since DTW has been highly appropriate for time series similarity measurement, several lower bounds have been proposed to speed up its expensive computation [1, 14]. Variants of DTW include cDTW [7, 20], EDR [3], and ERP [2]. A common characteristic of these methods is that they allow aligning elements "warped" in time. ERP and EDR satisfy the triangle inequality, and ERP is found to be more robust to noise than EDR. Another very recently proposed measure is MSM [21]. Applying in some sequence three operations (Move, Split, Merge) on a time series MSM is capable of transforming it into any other time series. MSM is metric, and it is also invariant to the choice of origin as opposed to ERP. Other proposed distance/similarity measures include Edit distance for strings and time series (e.g., for music retrieval [9]), LCSS [22], DISSIM [5], TWED [17], and SMBGT [9, 11].

2.2 Time Series Representations DFT is one of the most common representations, while some other more recent time series representations that capture global or local structural characteristics include SpaDe [4], and SAX [15]. Moreover, *Shapelets* [25] and variants [16] focus on determining discriminant time series subsequence patterns, and have been used for classification. For a comparison of several representations, please refer to Wang et al. [24], which demonstrates that there is little difference among them. All these approaches target in representing each individual time series by taking advantage of its structure. In this work, however, we try to represent "groups" of time series that belong to the same class. We focus on HMMs, which model the underlying structure of sequences determining the relationships between their observations. Thus, HMMs have been applied to speech recognition [19], classification [18], and music retrieval [10]. Although HMMs may be computationally expensive for their training, once they are constructed they can be highly applicable to time series search. CRFs [13, 23] can be used for modeling temporal patterns. Nonetheless, here we target in representing groups of "homogeneous" sequences by identifying the relationships among their observations.

3 Hidden Markov Models

Let $X = (x_1, \ldots, x_{|X|})$ and $Y = (y_1, \ldots, y_{|Y|})$ be two time series of length $|X|$ and $|Y|$, respectively, where $x_i, y_j \in \mathbb{R}$, $\forall(i = 1, \ldots, |X|; j = 1, \ldots, |Y|)$. A collection of N training time series is denoted as $\mathcal{D} = \{X_1, \ldots, X_N\}$. Given a distance measure $dist_x$, the distance between X and Y is a function $d_{\text{dist}_x}(X, Y)$.

An HMM is a doubly stochastic process containing a finite set of states [19]. Formally, it is defined by M distinct states, L values that can be observed at each state, the set $T = \{t_{uv}\}$ of transition probabilities, where $t_{uv} = P[s_t = v | s_{t-1} = u]$ with $1 \leq u, v \leq M$ and s_t being the state at time t (first order Markov chain), the set $E = \{e_v(k)\}$ of the probabilities of values at state v, where $e_v(k) = P[o_t = k | s_t = v]$ with o_t being the observed/emitted value at time t, and the set $\Pi = \{\pi_v\}$ of prior probabilities, where $\pi_v = P[s_1 = v], 1 \leq v \leq M$.

3.1 Training Assume that we have a dataset $\mathcal{D}$ with z classes $C_1, \ldots, C_z$. Let $\mathcal{C}_i$ be the set of training time series that belong to class C_i, with $i = 1, \ldots, z$. The size of $\mathcal{C}_i$ is denoted as $|\mathcal{C}_i| = n_i$. The training phase of an HMM for each $\mathcal{C}_i$ is split to two phases: a) initialization and b) iterative refinement.

Initialization Step: For each time series X_j $(j = 1, \ldots, n_i)$ of $\mathcal{C}_i$ $(i \in [1, z])$ we compute the average distance of all other time series $X_k \in \mathcal{C}_i$ $(j \neq k)$ to X_j. We choose DTW to be the distance measure, i.e., $dist_x = \text{DTW}$, since it has been shown to be one of the most competitive measures for time series matching [24]. In addition, we keep track of the *warping path* of all the pair-wise time series alignments involved in this process.

Next, we identify the *medoid* of $\mathcal{C}_i$, denoted as $X_{\mu_{C_i}}$, which is the time series with the minimum average distance to the rest of the training set. $X_{\mu_{C_i}}$ is broken into M equal-sized *segments*, where each segment $m \in [1, M]$ corresponds to one HMM state. Using these segments and the stored warping paths

we can determine the observed values of each state. Specifically, for each state (corresponding to a segment m) the observed values include all elements of $X_{\mu_{C_i}}$ in m, along with the elements of all time series in $\mathcal{C}_i$ that have been aligned to elements of m; the latter can be retrieved from the stored warping paths.

A common case in HMMs is to have for each state a Gaussian distribution for E, which is defined by the mean and standard deviation of the stored elements. To compute T, since we consider each segment as a state, at time t when an observation is emitted we can either stay at the same state or move forward to the next state. Let $|s_t|$ denote the total number of elements at time t of state s_t. The probability of jumping to the next state is $p = n_i/|s_t|$, and the probability of staying at the last state is 1. The complexity of this step is $O((n_i max_{j\in[1,n_i]}|X_j|)^2)$.

Iterative Refinement Step: In this step, we refine the z HMMs constructed during initialization. For a specific class C_i, for each $X_j \in \mathcal{C}_i$, we compute the Viterbi algorithm [19] to find its best state sequence. Specifically, let us denote as δ the Mx$|X_j|$ probability matrix. Each X_j always starts from state 1, and in the initialization phase the log-likelihood of its first element is computed according to the Gaussian distribution. The remaining elements of the first column of δ are set to $-\infty$. Since to get an observation we have either stayed at the same state or have performed a transition from the previous state, in the recursion phase we consider only the values of δ representing the probabilities of the previous element for the previous and the current state. For cell (u, v) these values are $\delta(u, v-1)$ and $\delta(u-1, v-1)$, which were computed in the initialization step. Hence, we first find the most probable transition by computing $m = max(\delta(u, v-1)+\log(t_{uu}), \delta(u-1, v-1) + \log(t_{u-1u}))$, and then $\delta(u, v) = m + \log(e_u(k))$. Finally, we backtrack from $\delta(M, |X_j|)$ and store the elements of X_j falling within each state. Having done this step for all $X_j \in \mathcal{C}_i$, the mean and standard deviation for E of each state, and also T are updated. The complexity of the aforementioned procedure is $O(M\sum_{j=1}^{n_i}|X_j|)$.

The refinement step is performed for the z HMMs and is repeated until the classification accuracy on the N training time series composing $\mathcal{D}$ cannot be further improved. The final outcome of this step is a set of z HMMs, denoted as $\mathcal{H} = \{H_1, \ldots, H_z\}$, where each $H_i \in \mathcal{H}$ defines a probability distribution for $\mathcal{C}_i$, which essentially describes the likelihood of observing any time series of class C_i.

Since we have created a "new" probabilistic space for time series similarity matching, we should define a way of measuring how "good" an HMM model $H_i \in \mathcal{H}$ is for a time series Q. This can be achieved by applying the Forward algorithm [19], which computes the likelihood of Q having been produced by H_i. Thus, the "goodness" of H_i is the likelihood estimate given by the algorithm. The complexity of the Forward algorithm is $O(|Q|M^2)$.

4 Experiments

4.1 Experimental Setup In this section, we present the setup for the model-based and distance-based time series search comparison.

Datasets: We experimented on the 45 time series datasets available from the UCR archive [8]. More details for each dataset can be found in Table 1.

Methods: We compared HMMs and four distance measures. Although any measure (Section 2.1) can be used to perform time series similarity search, an exhaustive consideration of all distance measures is practically impossible and beyond the scope of this paper. Here, we investigated DTW, cDTW with Sakoe-Chiba band [20], ERP, and MSM. These are not claimed to be the best measures to be used for any application domain. The rationales for selecting them are the following: (1) DTW is extensively used in time series and has been shown to provide excellent classification accuracy results [24], (2) cDTW is a much faster version of DTW providing equally good or better results than DTW [24], (3) ERP is essentially a variant of DTW and Edit Distance fixing the non-metric property of DTW, and (4) MSM is a recently proposed distance measure, which is metric and has been shown to outperform ERP and DTW in terms of NN classification accuracy on several datasets. The time complexity of all measures is quadratic, except for cDTW, which depends on the band constraint value.

Evaluation Measures: We evaluated the performance of HMMs against DTW, cDTW, ERP, and MSM on the training sets in terms of *classification error rate*. This rate is defined as the percentage of training time series misclassified using the Nearest Neighbor classifier. For the HMMs, each training time series is evaluated with all HMMs representing the classes of a dataset using the Forward algorithm, and the class of the HMM yielding the highest probability is considered to be the result of the classifier. To evaluate the model and distance-based time series search methods we used *precision* and *recall* for accuracy, and runtime for efficiency. Precision is the percentage of correct answers we get from the top-K results returned, while recall is the percentage of correct answers out of the total number of correct answers when the top-K results are returned. The goal is to achieve high precision and recall for small K.

With regard to the distance-based search methods, for each dataset, for each class we used all of its

training time series as queries (for a total of 15,741 queries for all datasets and classes), and found the DTW, cDTW, ERP, and MSM of all test time series to these queries (in total 61,691 test time series for all datasets). Regarding the model-based search method, we used each of the 423 trained models as a query and computed, for each dataset, the probability of each time series of the test set being produced by that query model. For both types of search methods, after computing the distances/probabilities, we sorted them in ascending/descending order and identified the ranks of the test time series that belong to the same "correct" class with that represented by each query. This was done in order to be able to compute the precision and recall. We note that for the distance-based methods, for each class, since there are many query time series we took the average precision and recall. All distance measures and the Forward algorithm were implemented in Java. Experiments were performed on a PC running Linux, with Intel Xeon Processor at 2.8GHz.

4.2 Experimental Results Next, we present our experimental findings.

4.2.1 Classification accuracy of HMMs In Table 1 we show for each of the 45 datasets the classification error rates attained on the training set for MSM, DTW, cDTW, ERP, and HMMs. The percentage of time series length $|X|$ defining M and the number of iterations to yield the error rate for HMMs, along with the value of the c parameter for MSM are also presented. We observe that HMMs achieve better than or equal error rate to that of the competitor distance measures in 32 datasets, out of which they outperform them in 28. The performance of HMMs is in many cases significantly better than all competitors. For example, for *ECGFiveDays* HMMs achieve an error rate of 0% as opposed to the next best which is 17.39%, while for 18 datasets (e.g., *50Words*, *Lightning-7*, *Adiac*, *Beef*, *OliveOil*, and *ECG_torso*) the error rate of HMMs is at least two times lower than that of the competitors. These numbers show that modeling time series classes with HMMs is highly competitive and promising for searching time series databases.

4.2.2 Precision - Recall In Figures 1, 2, 3, 4, 5, 6, 7, and 8, we present the average precision-recall of all classes (out of 423), which have at least 1, 10, 60, 120, and 200, less than 5 and 10, and between 10 and 15 training time series. In the figures we also show the number of classes of all datasets that satisfy the aforementioned thresholds ("num selected classes"), and also the total number of training time series of these selected classes ("sum training t.s."). DTW is referred to as "Unconstrained DTW" and cDTW as "Constrained DTW".

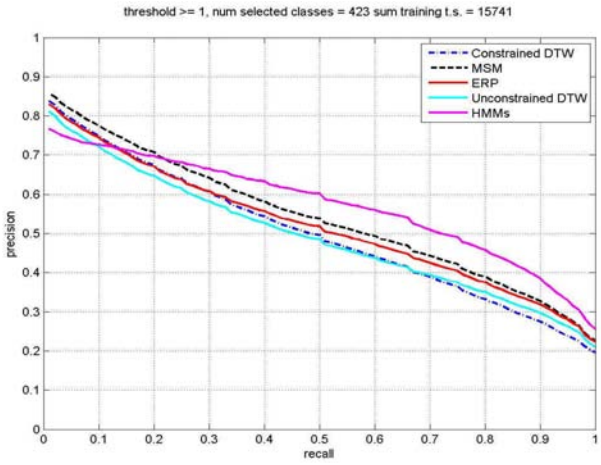


Figure 1: Precision vs Recall for model and distance-based time series search, when the number of training time series of the classes is >= 1.

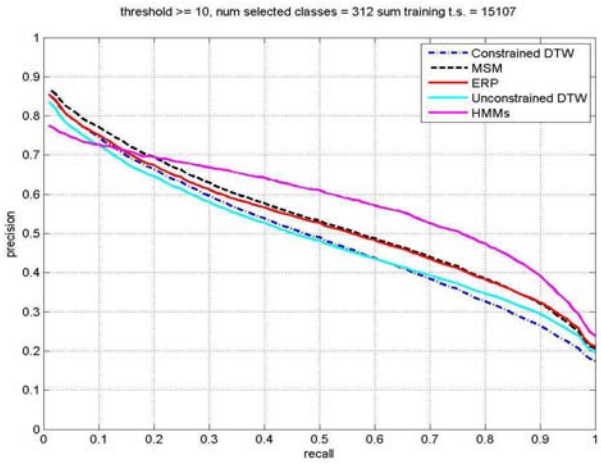


Figure 2: Precision vs Recall for model and distance-based time series search, when the number of training time series of the classes is >= 10.

In Figures 1 (i.e., all 423 classes have been considered in the displayed curves) and 2 we observe that the precision-recall curve for model-based search is not worse than all other distance-based methods in 9% recall, while it is better than all competitors in 20% recall. In Figure 3 HMMs method is best in 5% recall, showing that when there are many time series to train the HMMs, the model-based method performs noticeably better than the distance-based search methods. On the

Table 1: Nearest Neighbor classification error rates attained by MSM, DTW, cDTW, ERP, and HMMs on the training set of 45 datasets from the UCR repository of time series datasets. The table shows for each dataset: the number of training and test objects, the length of each time series in the dataset, the number of classes, the value of parameter c used by MSM on that dataset that yielded the lowest error rate on the training set (when two or three values are given, the one in italics was randomly chosen), the number of states as a percentage of the time series length and the number of iterations for which the HMMs achieved the lowest error rate on the training set. The numbers in bold indicate the smallest error rate.

ID	Dataset	train error rate (%)					train size	test size	length $\|X\|$	class num. z	parameter c (MSM)	state perc.	iter. num.
		MSM	DTW	cDTW	ERP	HMMs							
1	*Synthetic*	1.33	1.00	**0.33**	0.67	**0.33**	300	300	60	6	0.1	0.4	3
2	*CBF*	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	30	900	128	3	0.1	0.5	2
3	*FaceAll*	**1.07**	6.79	4.64	2.50	1.25	560	1,690	131	14	1	0.5	11
4	*OSU*	19.50	33.00	25.00	30.50	**17.00**	200	242	427	6	0.1	0.3	11
5	*SwedishLeaf*	12.40	24.60	18.00	13.40	**12.20**	500	625	128	15	1	0.9	6
6	*50Words*	21.11	33.11	23.33	28.22	**8.89**	450	455	270	50	1	0.5	1
7	*Trace*	1.00	**0.00**	1.00	9.00	**0.00**	100	100	275	4	0.01	0.3	2
8	*TwoPatterns*	**0.00**	**0.00**	0.20	**0.00**	**0.00**	1,000	4,000	128	4	1	0.1	1
9	*FaceFour*	8.33	25.00	12.50	12.50	**0.00**	24	88	350	4	1	0.1	1
10	*Lightning-7*	27.14	32.86	20.00	28.57	**7.14**	70	73	319	7	1	0.2	1
11	*Adiac*	38.97	40.51	39.74	39.49	**15.90**	390	391	176	37	1	0.5	8
12	*Fish*	13.71	26.29	22.86	17.14	**7.43**	175	175	463	7	0.1	0.5	4
13	*Beef*	66.67	53.33	50.00	66.67	**23.33**	30	30	470	5	0.1	0.4	2
14	*OliveOil*	16.67	13.33	10.00	16.67	**3.33**	30	30	570	4	0.01	0.3	2
15	*ChlorineConc.*	38.97	38.97	**36.62**	38.76	56.32	467	3,840	166	3	1	0.7	4
16	*ECG_torso*	12.50	32.50	7.50	25.00	**2.50**	40	1,380	1,639	4	1	0.4	3
17	*Cricket_X*	18.46	20.26	**17.18**	21.54	25.38	390	390	300	12	1	0.5	14
18	*Cricket_Y*	24.10	20.51	**18.21**	23.33	19.23	390	390	300	12	0.1, *1*	0.4	9
19	*Cricket_Z*	24.10	22.56	**17.69**	24.87	23.08	390	390	300	12	1	0.4	14
20	*Diatom Red.*	6.25	6.25	6.25	6.25	**0.00**	16	306	345	4	0.01, 0.1, *1*	0.1	4
21	*FacesUCR*	2.50	10.00	8.00	5.50	**0.50**	200	2,050	131	14	1	0.8	9
22	*Haptics*	49.68	58.71	46.45	54.19	**29.68**	155	308	1,092	5	1	0.1	15
23	*InlineSkate*	50.00	59.00	58.00	49.00	**43.00**	100	550	1,882	7	1	0.5	3
24	*MALLAT*	5.45	5.45	1.82	5.45	**0.00**	55	2,345	1,024	8	1	0.1	1
25	*MedicalImages*	27.82	27.56	**26.25**	26.51	34.91	381	760	99	10	0.1	0.4	13
26	*StarLightC.*	10.70	9.60	9.30	13.80	**8.60**	1,000	8,236	1,024	3	0.1	0.5	14
27	*Symbols*	**0.00**	4.00	4.00	8.00	**0.00**	25	995	398	6	0.1	0.1	1
28	*uWaveGest_X*	25.78	29.35	**24.89**	26.67	25.22	896	3,582	315	8	0.1, *1*	0.5	11
29	*uWaveGest_Y*	28.24	37.05	**27.57**	33.93	34.60	896	3,582	315	8	1	0.7	13
30	*uWaveGest_Z*	29.24	33.59	30.13	31.25	**27.46**	896	3,582	315	8	1	0.2	12
31	*WordsSynon.*	22.10	36.33	27.34	28.46	**13.86**	267	638	270	25	1	0.8	13
32	*ECGThorax1*	18.17	20.11	18.11	17.83	**7.17**	1,800	1,965	750	42	1	0.5	15
33	*ECGThorax2*	10.83	14.17	12.44	11.72	**7.67**	1,800	1,965	750	42	1	0.5	14
34	*Gun Point*	**4.00**	18.00	**4.00**	8.00	8.00	50	150	150	2	0.01	0.3	2
35	*Wafer*	**0.10**	1.40	0.30	**0.10**	1.70	1,000	6,164	152	2	1	0.9	4
36	*Lightning-2*	16.67	13.33	10.00	13.33	**5.00**	60	61	637	2	0.01	0.1	15
37	*ECG*	14.00	23.00	14.00	18.00	**12.00**	100	100	96	2	1	0.8	2
38	*Yoga*	**12.00**	18.33	17.67	17.33	22.33	300	3,000	426	2	0.1	0.2	15
39	*Coffee*	25.00	**14.29**	**14.29**	25.00	21.43	28	28	286	2	0.01	0.3	1
40	*ECGFiveDays*	26.09	43.48	17.39	26.09	**0.00**	23	861	136	2	1	0.2	4
41	*ItalyPowerDemand*	**4.48**	**4.48**	**4.48**	5.97	5.97	67	1,029	24	2	0.1, *1*	0.9	2
42	*MoteStrain*	15.00	25.00	25.00	25.00	**0.00**	20	1,252	84	2	0.1	0.5	7
43	*SonySurfaceI*	10.00	20.00	10.00	15.00	**0.00**	20	601	70	2	1	0.1	1
44	*SonySurfaceII*	11.11	14.81	14.81	18.52	**3.70**	27	953	65	2	0.1	0.3	1
45	*TwoLeadECG*	4.35	8.70	8.70	4.35	**0.00**	23	1,139	82	2	*0.01*, 0.1	0.1	4

contrary, when the number of training time series gets extremely high, i.e., at least 120 or (much worse) 200 (Figures 4 and 5), the model-based method is not better than the competitors when recall is around 17% and 33%, respectively. In the latter figure we also observe that the curves for DTW and ERP-based methods are very close to that of model-based search. The main reason for the last effect is that the curve for HMMs includes the results from classes with more than 300 training time series, i.e., the third class of dataset 26 that has 573 and the second class of dataset 35 that has 903 training time series (1,476 in total), consisting the 9.37% of the total number of training time series of all datasets. As a result, there has been overfitting for these HMMs, which influences their performance when searching time series datasets. However, it is important to note that, as the threshold increases (Figures 1 - 5), the precision is getting higher as the recall increases, which is essentially what we target for when searching time series databases. Referring to the distance-based methods, in Figures 1 and 2 the worst performance is presented by DTW and cDTW, while in Figures 3, 4, and 5 cDTW and MSM present the worst curve among all methods.

In Figures 6 and 7 we show that the precision-recall curve for HMMs is particularly influenced by the number of time series used to train them. In Figure 6 the curve corresponds to the average precision-recall of all classes/models that have less than 5 training time series, while including classes with more training time series, up to 10 for Figure 7 the curve is much better. In both figures the most competitive distance-based method is the MSM. Finally, in Figure 8 we present the precision-recall of all methods for all classes that have at least 10 and up to 15 training time series. Clearly, we can see that the HMMs perform best for any recall value, showing that when the HMMs have been constructed exploiting a sufficient (and concurrently not overwhelm-

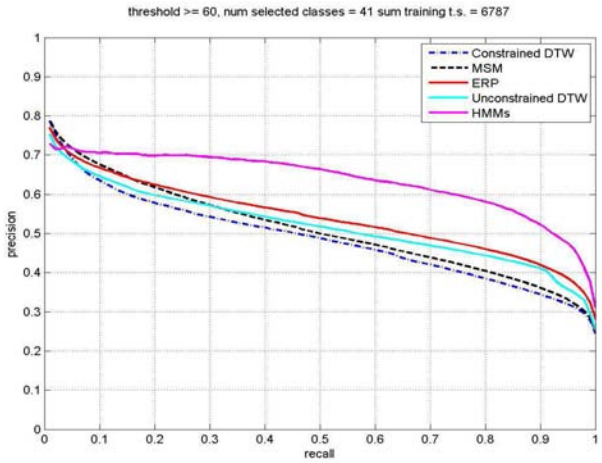


Figure 3: Precision vs Recall for model and distance-based time series search, when the number of training time series of the classes is >= 60.

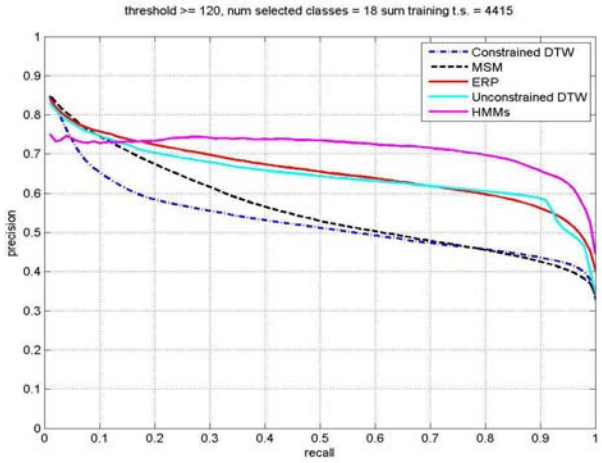


Figure 4: Precision vs Recall for model and distance-based time series search, when the number of training time series of the classes is >= 120.

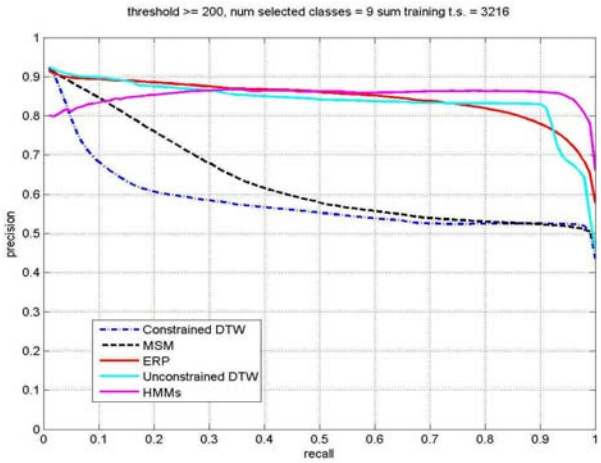


Figure 5: Precision vs Recall for model and distance-based time series search, when the number of training time series of the classes is >= 200.

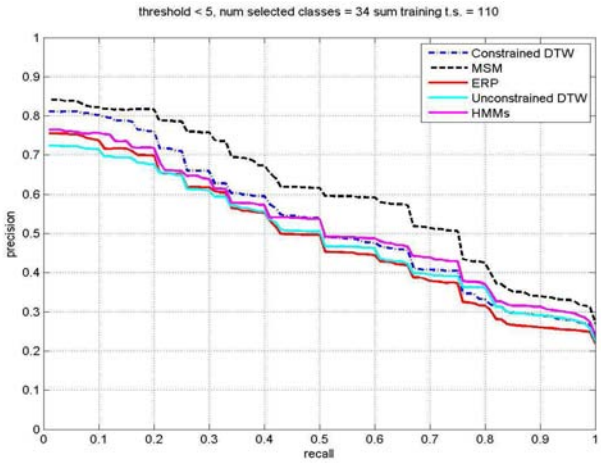


Figure 6: Precision vs Recall for model and distance-based time series search, when the number of training time series of the classes is < 5.

ing) number of training time series comprising their classes, the probability distribution of such classes effectively represents their (and similar) time series.

4.2.3 Efficiency In Table 2, for the distance-based methods we present, for each dataset, the average time to find the distance of a training time series/query with all test time series, and for the model-based method we present the average time per model to compute the Forward algorithm with all test time series. According to the Table, cDTW method is always faster than the competitor methods, except for dataset with ID 25 where ERP is the fastest method. We also observe ERP is 1.5 to 2 time faster than MSM, which is always faster than DTW. In addition, model-based search is basically slower than all competitors, except for datasets with ID 8 (it is better than DTW), 9 (worse than only cDTW), and 20, 22, 24, 27, 36, 43, and 45 (worse than only cDTW and ERP). The explanation for these numbers is that the Forward algorithm is more time consuming than the distance measures. We also have to note that in several datasets the differences in runtimes are not clearly noticeable among the competitors.

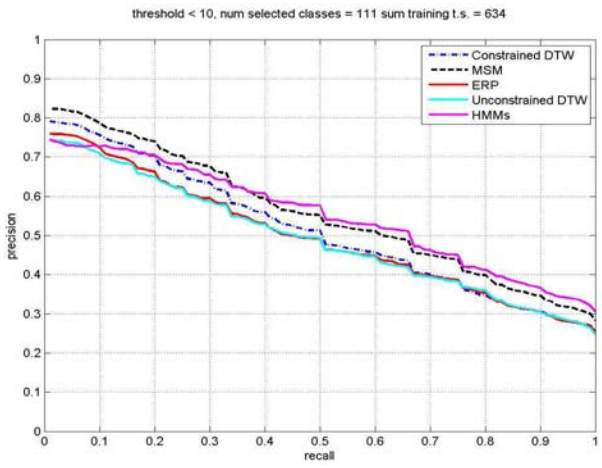


Figure 7: Precision vs Recall for model and distance-based time series search, when the number of training time series of the classes is < 10.

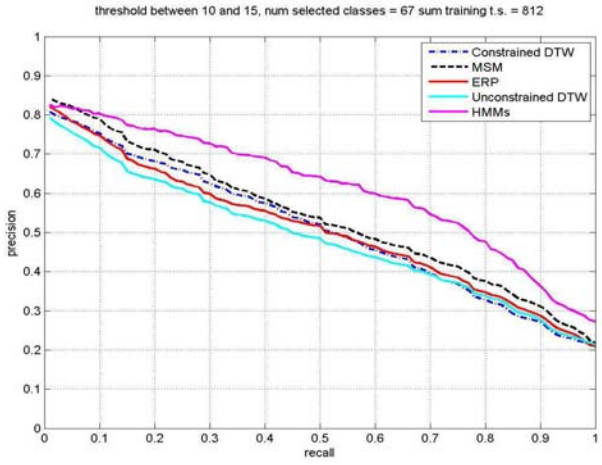


Figure 8: Precision vs Recall for model and distance-based time series search, when the number of training time series of the classes is between 10 and 15.

5 Conclusions and Future Work

In this paper, we presented an effective way of modeling classes of time series via HMMs, which have been shown to be highly applicable to a variety of domains. Comparing the classification error rates of HMMs and four distance measures (DTW, cDTW, MSM, ERP) on the training sets of 45 datasets we observed that HMMs achieve better than or equal error rates to those of the competitors on 32 datasets. Furthermore, evaluating the model-based search with the four distance-based methods on the 45 datasets we noticed that when the number of training time series is much less than 10, the performance of model-based search is not robust and this type of search is outperformed by MSM. However, when a sufficient number of time series is provided to train the HMMs, i.e., more than 10, which is usually the case in the 423 classes of these datasets, the model-based

Table 2: Average time (in seconds) of DTW, cDTW, MSM, ERP, and HMMs for searching on each of the 45 datasets from the UCR repository of time series datasets.

ID	DTW	cDTW	MSM	ERP	HMMs
1	0.102	0.03	0.09	0.048	0.222
2	1.236	0.414	1.08	0.552	3.534
3	2.466	0.396	2.148	1.074	6.888
4	3.438	0.786	2.724	1.506	9.648
5	0.828	0.126	0.696	0.366	6.678
6	2.568	0.564	2.076	1.134	7.74
7	0.6	0.096	0.492	0.258	1.08
8	5.586	1.068	4.62	2.472	3.018
9	0.864	0.12	0.708	0.372	0.504
10	0.588	0.114	0.498	0.258	0.684
11	0.954	0.156	0.75	0.42	2.856
12	2.844	0.504	2.19	1.26	8.736
13	0.51	0.048	0.408	0.24	1.29
14	0.768	0.084	0.612	0.336	1.35
15	9.042	0.894	7.848	3.9	58.074
16	343.44	77.646	283.536	187.506	732.216
17	2.754	0.666	2.388	1.218	8.388
18	2.79	1.164	2.376	1.2	6.864
19	2.79	0.654	2.382	1.206	6.864
20	2.868	0.27	2.232	1.206	1.65
21	3.06	1.008	2.634	1.32	13.722
22	30.246	5.394	23.814	13.446	16.488
23	179.946	81.78	152.154	99.09	483.348
24	202.77	23.928	158.07	88.728	111.804
25	0.708	0.306	0.516	0.276	1.446
26	717.522	295.884	544.434	322.932	2082.708
27	12.33	3.03	9.396	5.1	7.068
28	27.864	4.668	20.874	11.526	84.786
29	27.906	4.722	21.06	11.586	118.428
30	27.84	5.652	20.88	11.556	33.024
31	3.648	0.894	2.922	1.53	18.156
32	89.34	11.346	70.728	38.568	257.526
33	89.166	11.478	71.064	38.946	259.422
34	0.264	0.03	0.21	0.12	0.174
35	11.382	1.404	8.88	4.992	62.634
36	1.998	0.438	1.614	0.87	1.134
37	0.078	0.012	0.066	0.036	0.384
38	42.756	5.502	33.132	18.69	50.4
39	0.18	0.03	0.15	0.084	0.336
40	1.362	0.15	1.11	0.57	1.518
41	0.084	0.042	0.078	0.054	0.312
42	0.75	0.114	0.63	0.342	2.19
43	0.27	0.048	0.234	0.12	0.168
44	0.492	0.066	0.33	0.174	0.606
45	0.696	0.162	0.522	0.294	0.402

method provides better tradeoffs between precision and recall than the competitors. Regarding runtime, model-based search is usually slower than the competitors due to the demanding Forward algorithm.

As the aforementioned accuracy results indicate, HMMs are particularly useful for searching time series databases. Hence, for future work, we plan to develop a GUI for interactive data analysis, where the user will be able to search time series databases and compare the performance of distance-based and model-based methods. Specifically, the user will be able to perform the following actions: (a) see the training time series belonging to each class of several datasets, (b) choose the distance measures (out of a pool of measures) for evaluation and comparison with model-based search, (c) set the threshold and choose the datasets (or even classes) to be considered, (d) select the time series, and also the number of states and iterations to individually train each HMM, instead of forcing all HMMs of a dataset to have the same values for the last two parameters; by doing so, the user will be able to further improve the precision-recall curves. Finally, the user will be able to interact with the resulting figures, e.g., by getting the exact values for precision and recall for each method when pointing on its curve.

References

[1] I. Assent, M. Wichterich, R. Krieger, H. Kremer, and T. Seidl. Anticipatory dtw for efficient similarity search in time series databases. *PVLDB*, 2(1):826–837, 2009.

[2] L. Chen and R. Ng. On the marriage of l_p-norms and edit distance. In *VLDB*, pages 792–803, 2004.

[3] L. Chen and M. T. Özsu. Robust and fast similarity search for moving object trajectories. In *SIGMOD*, pages 491–502, 2005.

[4] Y. Chen, M. A. Nascimento, B. Chin, O. Anthony, and K. H. Tung. Spade: On shape-based pattern detection in streaming time series. In *ICDE*, pages 786–795, 2007.

[5] E. Frentzos, K. Gratsias, and Y. Theodoridis. Index-based most similar trajectory search. In *ICDE*, pages 816–825, 2007.

[6] J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques*. Morgan kaufmann, 2006.

[7] F. Itakura. Minimum prediction residual principle applied to speech recognition. *Transactions on Acoustics, Speech and Signal Processing*, 23(1):67–72, 1975.

[8] E. Keogh, Q. Zhu, B. Hu, Y. Hao, X. Xi, L. Wei, and C. Ratanamahatana. The UCR time series classification/clustering homepage: www.cs.ucr.edu/~eamonn/time_series_data/, 2011.

[9] A. Kotsifakos, P. Papapetrou, J. Hollmén, and D. Gunopulos. A subsequence matching with gaps-range-tolerances framework: A query-by-humming application. *PVLDB*, 4(11):761–771, 2011.

[10] A. Kotsifakos, P. Papapetrou, J. Hollmén, D. Gunopulos, and V. Athitsos. A survey of query-by-humming similarity methods. In *PETRA*, pages 5:1–5:4, 2012.

[11] A. Kotsifakos, P. Papapetrou, J. Hollmén, D. Gunopulos, V. Athitsos, and G. Kollios. Hum-a-song: a subsequence matching with gaps-range-tolerances query-by-humming system. *PVLDB*, 5(12):1930–1933, 2012.

[12] J. B. Kruskall and M. Liberman. The symmetric time warping algorithm: From continuous to discrete. In *Time Warps*. Addison-Wesley, 1983.

[13] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289, 2001.

[14] D. Lemire. Faster retrieval with a two-pass dynamic-time-warping lower bound. *Pattern recognition*, 42(9):2169–2180, 2009.

[15] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *SIGMOD workshop DMKD*, pages 2–11, 2003.

[16] J. Lines, L. M. Davis, J. Hills, and A. Bagnall. A shapelet transform for time series classification. In *SIGKDD*, pages 289–297, 2012.

[17] P.-F. Marteau. Time warp edit distance with stiffness adjustment for time series matching. *Pattern Analysis and Machine Intelligence*, 31(2):306–318, 2009.

[18] A. Pikrakis, S. Theodoridis, and D. Kamarotos. Classification of musical patterns using variable duration hidden Markov models. *Transactions on Audio, Speech, and Language Processing*, 14(5):1795–1807, 2006.

[19] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[20] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Transactions on Acoustics, Speech and Signal Processing*, 26:43–49, 1978.

[21] A. Stefan, V. Athitsos, and G. Das. The move-split-merge metric for time series. *Transactions on Knowledge and Data Engineering*, 2012.

[22] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *ICDE*, pages 673–684, 2002.

[23] S. B. Wang, A. Quattoni, L.-P. Morency, D. Demirdjian, and T. Darrell. Hidden conditional random fields for gesture recognition. In *CVPR*, pages 1521–1527, 2006.

[24] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. J. Keogh. Experimental comparison of representation methods and distance measures for time series data. *Data Minining and Knowledge Discovery*, 26(2):275–309, 2013.

[25] L. Ye and E. Keogh. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data Mining and Knowledge Discovery*, 22(1-2):149–182, 2011.

In-depth Interactive Visual Exploration for Bridging Unstructured and Structured Document Content

Axel J. Soto* Ryan Kiros† Vlado Keselj* Evangelos Milios*

Abstract

Semi-structured data refers to the combination of unstructured and structured data. Unstructured data is free text in natural language, while structured data is typically stored in tables and following a data schema. Recent statistics shows that 80% of the data generated in the last two years is unstructured. However, one interesting observation is that free text usually comes along with some structured data, or meta-data describing or adding more information about the text. In this paper we present ViTA-SSD, a Visual Text Analytics Tool for Semi-Structured Data. This tool aims at extracting interesting patterns in semi-structured data through the joint consideration of the free text and the meta-data. This represents a challenging task because an effective approach needs the combined effort of text mining algorithms and human experts who can drive the exploration process in a meaningful way. A related challenge is the appropriate visualization and understanding of the patterns found. In order to address these challenges, our visual analytics tool takes advantage of a novel dimensionality reduction and a fast user-supervised clustering method. We showcase our tool here as well as we reflect on some lessons learned from the development and evaluation of our tool.

1 Introduction

Even though research in machine learning and text mining has contributed to the development of methods for automatic document analysis, finding actionable insight in text data usually requires a human user that queries and interacts with the data in order to find answers to her questions. Yet the field has so far not fully developed human-in-the-loop approaches to enable the user to perform visual data exploration and visual analytics, so that the user may arrive at meaningful and relevant results [3].

We will use the term *semi-structured data* to refer to the data that have an unstructured component (free text) and a set of structured fields associated with each document (meta-data). Despite the fact that semi-structured data can be found in many scenarios, little work has been done on analyzing information from semi-structured data. Kandel et al. [2] report interviews with 35 data analysts, where they indicate the processing of semi-structured data as one of the major analysts' challenges.

For instance, an analyst aided by a text mining tool may find several clusters of documents describing problems that are similar or related. However, the analyst would also like to relate this information to the meta-data present in the documents. In the context of aviation incident reports[1], some possible questions an analyst may ask include: are certain incident descriptions more likely to be produced in certain locations than others? To answer this type of questions, or to even discover unexpected relationships in our data, we need computational support to bridge the analysis of the structured and unstructured components of the data.

In this work, we present a Visual Text Analytics Tool for Semi-Structured Documents (ViTA-SSD). The system has two main user-oriented goals. First, to enable a user to discover patterns hidden in the text and meta-data by means of a visual and interactive exploration of the data. This is done by combining the results of a user-driven document clustering with the associated document meta-data. Second, to make the discovered patterns understandable and verifiable by the users. To address these goals we have developed a novel dimensionality reduction method, which is one of the core elements of the system to represent the unstructured data internally. We have also redesigned a clustering approach for large document corpora that allows user supervision. We will give a brief overview of the main components of this system here, and we will demo the system during the workshop. We conclude this paper formulating general design principles that aim at being applicable to any other visual text analytics systems.

2 Statistical and Data Mining Techniques

2.1 Dimensionality reduction methods on text generate a new (and typically smaller) set of latent

*Dalhousie University, Canada – soto@cs.dal.ca
†University of Toronto, Canada

[1] We use the ASRS data for examples this paper `http://asrs.arc.nasa.gov/`

features that can capture richer semantic relationships. This compact representation serves two other purposes. First, it provides a two-dimensional representation of the corpus that users can interact with. Second, since the reduced space is a good approximation of the raw representation of the corpus, this compact representation allows measuring similarity among documents considerably faster, and hence clustering algorithms can be faster too.

Numerous approaches have used variants of Principal Component Analysis (PCA), Latent Semantic Indexing, or Multi-Dimensional Scaling as dimensionality reduction methods for text. The main problem with PCA-like approaches and classical scaling methods is that they focus mainly on retaining large pairwise distances, which is undesirable for document corpora because distances between similar documents is not well preserved. The machine learning community has been active in the area of dimensionality reduction and feature learning, and many algorithms based on deep learning have been proposed [1]. Deep learning typically involves a greedy layer-by-layer training of bipartite Markov random fields known as Restricted Boltzmann Machines. This type of training could be used to learn feature hierarchies from data as a preprocessing step to learning tasks or for initialization of the weights of a deep neural network. These methods have been shown to lead to better embeddings than most popular non-linear dimensionality reduction methods [1]. While these methods have an expensive training phase, they can handle new incoming data efficiently.

ViTA-SSD incorporates a novel unsupervised dimensionality reduction method which we shall denote as *deep t-distributed correlative subspace mapping* (dt-CSM). Our algorithm uses a deep pre-trained network to learn a mapping by maximizing: $r(P, Q)$, where r is the Pearson correlation, and P and Q denote document pairwise similarities in the original and latent space, respectively. This objective function is inspired by [6]. In this way we can extend a linear dimensionality reduction method to learn a highly non-linear parametric mapping.

2.2 Fast user-supervised clustering. One important design choice in exploratory data analysis systems is given by the tradeoff between accuracy and speed of the text mining methods. Clustering is one crucial component of ViTA-SSD as it is used to support corpus exploration and sense-making [4]. Therefore, we have incorporated the clustering method proposed in [5], which was shown to have a significant speed-up with a marginal drop of the clustering accuracy compared to the traditional k-means.

Yet another important feature of visual analytics systems is to provide some sort of supervision to better reflect users' intentions. Similar to [7] and others therein, our approach makes use of feature supervision for influencing cluster formation. In these approaches, relevant terms are provided to the algorithm, so that they can have an impact when calculating distances among documents. Our approach differs from previous works, in the sense that the feature supervision strategy can be applied on a different feature space, e.g. after a non-linear dimensionality reduction step. In this way, we have adapted the fast clustering algorithm presented in [5] to modify distances to centroids based on the occurrence of the suggested terms without the need to compute distances in the original feature space.

3 ViTA-SSD Interface

ViTA-SSD aims at supporting the exploration of semi-structured data and helping to identify relevant patterns in the data. There are many different ways the data can be explored, such as searching, filtering or clustering. Clustering provides a way of aggregating related documents and get a sense about what they have in common by interacting with their main keywords and representative documents. Clusters are also a critical component for the correlation of related free-text descriptions with particular meta-data values. The interface of ViTA-SSD is divided in six coordinated panels. Due to space limitations we will just describe the two most important visualizations: the *document exploration* and the *meta-data analysis* panel.

Let us illustrate the clustering and its user-supervision process on the *document exploration*. The collection of documents at an initial stage is shown in Fig. 1(a). Let us assume that the analyst would like to have a cluster only devoted to bird strike incidents while keeping all other documents selected. We can see that cluster 4 (dark green) has the words *engine*, *emergency* and *landing* among the most representative ones. Note that *bird* is also listed, but with a smaller font. The result of searching for "bird" is shown in Fig. 1(b). We can see that most documents are a subset of the original cluster 4 with some others being close but in neighboring clusters. This illustrates the capacity of dt-CSM to locate related documents close to each other. The state of the system after "bird" is suggested as a relevant keyword (with influence factor at 10%) is shown in Fig. 1(c). We can note how cluster 4 has shrunk around the documents about bird strike incidents, and "bird" became the second largest word. The influence factor of the suggestion allows the user to control the extent that distances are adapted to cluster together documents containing the suggested term. Finally, Fig. 1(d)

shows the clustering assignment after supervision with the influence factor at 80%. We can now see that the cluster keywords better reflect bird strike incidents. The cluster has become much more exclusive and distances in that neighborhood are distorted to benefit documents that contain "bird". Pushing this influence factor even further (90%-100%) makes all documents that contain the word "bird" (Fig. 1(b)) members of the same cluster.

The meta-data analysis panel is responsible for bridging and visualizing patterns between the unstructured and structured content (Fig. 2). This visualization is organized as a matrix layout, where rows represent clusters and columns represent meta-data values for a given meta-data field. Each cell is colored orange and red according to the existence of strong or very strong evidence of an unlikely correlation, defined as $p < 0.05$ or $p < 0.01$, where p quantifies the probability of expecting this correlation from the data distribution. The intuition behind a colored cell is described in Fig. 2(b). There are some other information and interactions available in this visualizations, which we do not comment on due to space reasons.

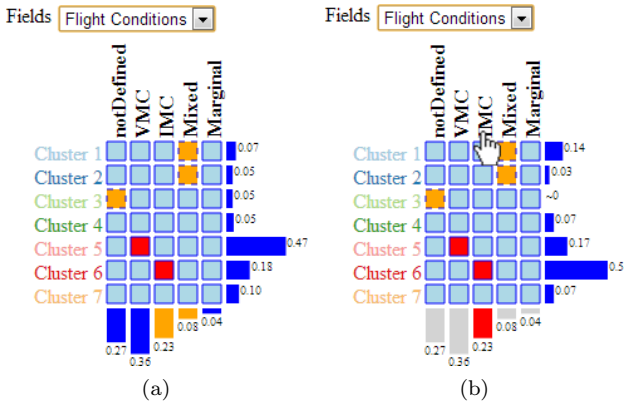


Figure 2: Meta-data analysis panel for "Flight conditions" after filtering incident reports about turbulence encounters. (a)Very strong correlations can be identified e.g. between "Cluster 6" and "IMC" (Instrument Meteorological Conditions). (b) When the meta-data value "IMC" is hovered over, we can see how incidents that happen under this flight category are assigned to the different clusters. The large increase in documents assigned to "Cluster 6" (from 18% to 50%) suggests that an interesting correlation happens between documents assigned to "Cluster 6" and the "IMC" category.

4 Discussion

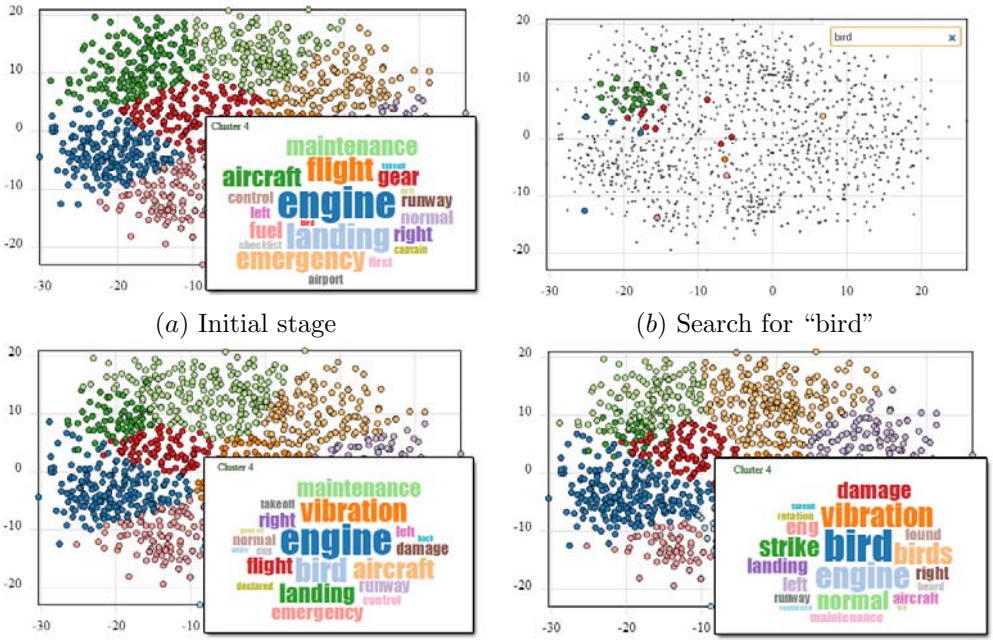
The development of our visual analytics tool allowed us to learn some lessons and to propose guidelines that are applicable to the design of any visual text analytics method:

Fast processing after interactions. Unfortunately many text mining and machine learning algorithms are compute intensive and the time needed to run may get prohibitive for large document collections or for real-time usage. In order to keep users engaged exploring in the document collection and interacting with the interface, it is important that the latency time between the interaction submission and its outcome can be as short as possible. This leads to a careful selection of algorithms that may be pre-computed off-line, and those that may be executable in real time. For the ones that need to run on-line, there is usually a clear trade-off between result precision and computation speed. In many scenarios a faster outcome with a slightly larger error may be more desirable than the slower but more accurate decision. One way to deal with this issue is to place this responsibility on the user by providing a way of manually controlling the computation time in tradeoff to its accuracy.

Simple method parameterization. Text mining methods usually require some sort of parameterization that makes the method more adaptable to different scenarios. However, we believe that domain experts tend to be reluctant to learn complex computing concepts if these are not strictly related to their work. The challenge here is to select methods that enable an intuitive parameterization for a lay person or design the interface in such a way that parameters are replaced by more user-oriented meta-parameters. Also, we found useful from our user study to provide a two-level option interface, where more advanced features are not visible but can be accessed on demand for increased flexibility.

Influencing similarity. Measuring similarity is key in text mining. An old and recurring question in text mining is what criteria should be followed to consider two documents to be similar to each other. There is no simple answer to this question as different users may regard different features to be relevant for similarity. Visual analytics can provide a way to deal with this issue as opposed to defining a fixed criterion or distance measure in advance. In this regard, we consider important to provide different channels for corpus exploration as well as different ways to influence the text mining algorithms.

One of the main characteristics of the modern phenomenon called *Big Data* is that documents are not only a large static repository, but also arrive as a never-ending stream of data. This imposes some limitations



(*a*) Initial stage (*b*) Search for "bird"

(*c*) *bird* suggested at 10% (*b*) *bird* suggested at 80%

Figure 1: Use case of the fast user-supervised clustering algorithm where the user is interested in obtaining one cluster about bird strike incidents.

on the methods we can use for dimensionality reduction on the document collections. Models based on deep architectures scale well with very large data sets because the model is trained in mini-batches of data, as opposed to other traditional dimensionality reduction methods that need to store all pairwise similarities, and hence suffer from a memory limitation. While models based on deep learning are slow to train, it is a one-time effort that can be performed off-line. Moreover, in the scenario of dynamic data, new incoming documents could be visualized or projected immediately by using the representation that was learnt based on previous documents.

ViTA-SSD successfully integrates novel machine learning methods with interactive visualizations to support exploration and pattern finding in semi-structured data. Numerical experiments have been carried out in order to assess and compare our dimensionality reduction method and fast user-supervised clustering algorithm. We have also run a user-study that allowed us to evaluate ViTA-SSD from a user-centered perspective.

Acknowledgements

This research has been supported by The Boeing Company and a grant from the Natural Sciences and Engineering Research Council of Canada.

References

[1] Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

[2] S. Kandel, A. Paepcke, J. M. Hellerstein, and J. Heer. Enterprise data analysis and visualization: An interview study. In *Visual Analytics Science and Technology, 2012. VAST 2012. IEEE Conference on*, 2012.

[3] D. A. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann. *Mastering the Information Age-Solving Problems with Visual Analytics*. Florian Mansmann, 2010.

[4] Y. Liu, S. Barlowe, Y. Feng, J. Yang, and M. Jiang. Evaluating exploratory visualization systems: A user study on how clustering-based visualization systems support information seeking from large document collections. *Information Visualization*, 12(1):25–43, 2013.

[5] D. Sculley. Web-scale k-means clustering. In *Proceedings of the 19th International Conference on World Wide Web*, pages 1177–1178. ACM, 2010.

[6] M. Strickert, A. J. Soto, and G. E. Vazquez. Adaptive matrix distances aiming at optimum regression subspaces. In *Proceedings of the European Symposium on Artificial Neural Networks*, pages 93–98. D-facto Publications, 2010.

[7] D. M. Witten and R. Tibshirani. A framework for feature selection in clustering. *Journal of the American Statistical Association*, 105(490), 2010.