Visualising the Evolution of Dynamic Communities in Social Networks using Timelines

Hugo Hromic and Conor Hayes

Insight Centre for Data Analytics @ NUI Galway
{first.last}@insight-centre.org
http://www.insight-centre.org/

Abstract. Real-world social networks from a variety of domains can naturally be modelled as dynamic graphs. However, approaches for detecting communities mostly focus on identifying them in static graphs. Researchers often consider the problem of tracking the evolution of communities in dynamic scenarios. In this work we present the *Dynamic Community Viewer* (DCV), a visualisation tool for tracking the life cycle of communities over time in a dynamic network, where each community is characterised by a series of significant evolutionary events. The DCV is capable of visualising the development of these events at different points in time using browsable timelines. Specifically, it can visualise the birth, death, split, merge, contraction, expansion and any user-defined attribute change (e.g. topics) as evolutionary events for sets of dynamic communities. Our tool is based on an established community tracking model that leverages a community-matching strategy for efficiently identifying and tracking dynamic communities.

Keywords: Social Networks · Dynamic Communities · Visualisation.

1 Introduction

Social network analysis is traditionally focused on the representation of graphs as static networks for specific time steps. This representation is widely adopted for the task of community detection, where the goal is to identify meaningful group structures in these networks. However, representing dynamic sources of data using such static networks can lead to group structures present in short periods of time to be difficult to identify or be completely unnoticed by the community detection approach. Discarding temporal information can lead to loosing the detail of the evolutionary behaviour of these groups.

One of the most interesting aspects of community detection for researchers and decision makers is the ability to visualise how the user communities under study are evolving in time. However, most community finding approaches are unable to generate such view directly, but instead focus on identifying static sets of communities at particular points in time. Fortunately, dynamic user community tracking algorithms exist [8] that can analyse arbitrarily discovered communities between consecutive time steps and identify similarities from a past set of communities to a current updated configuration.

Using the dynamic community tracking algorithm in [8] as a base, we build a visualisation tool capable of extracting eight types of evolutionary events from a set of communities discovered at consecutive time steps: (1) *birth*, (2) *death*, (3) *split*, (4) *merge*, (5) *expansion*, (6) *contraction*, (7) *intermittence*, and (8) *attribute change*.

Our proposed workflow can be seen in Figure 1. First, the user chooses a community detection approach suitable for the dataset under study. Then, sets of independent static communities $C_{t,i}$ are extracted for consecutive time steps t using any desired granularity, e.g. hourly or daily. With these static step communities, the user applies the dynamic community tracking approach from Greene et.al [8], called TRACKER, to obtain dynamic timelines that relate step communities to each other in time considering the birth, death, split, merge and intermittence evolutionary events. Finally, the user applies our post-processing algorithm, TRACKER2VIS, to compute the remaining evolutionary events, i.e. expansion, contraction and attribute change, and generate the complete visualisation information necessary for the interactive user interface. It is important to note that our tool is able to visualise any dynamic timeline computed using TRACKER, which mainly utilises the Jaccard similarity metric to compare "communities" across time steps. This characteristic makes their tracking approach highly convenient and flexible. In consequence, any group-like structure from any clustering technique can be used and despite the visualisation being designed towards community analytics, it is not strictly restricted only to this use case.

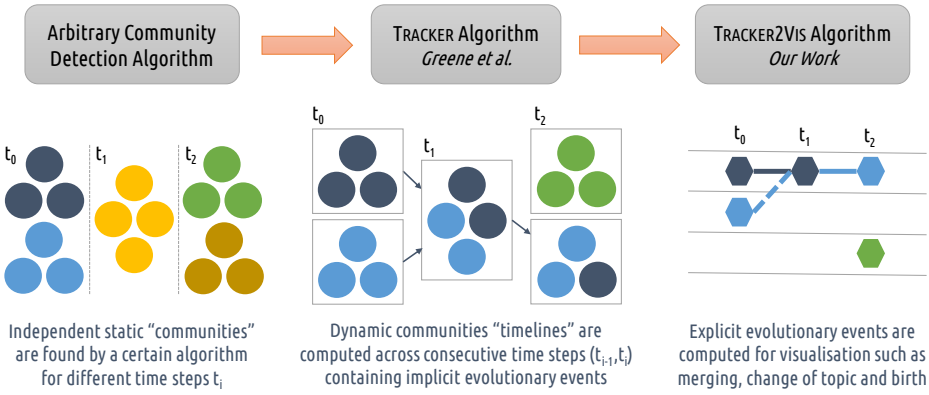


Fig. 1. Usage workflow for visualising dynamic communities using our approach.

To achieve all the above functionality, our work proposes three main contributions: (1) we build a post-processing algorithm on top of the work of Greene et al. [8] for extracting evolutionary events from dynamic communities, (2) we propose an additional event type (*attribute change*) to support visualisation of user-defined aspects such as topics, and (3) we develop a web-based visualisation tool, the *Dynamic Community Viewer* (DCV), that combines the above contributions into a concrete general purpose tool. The source code[1] and an interactive demo of the DCV[2] are both available online.

The rest of this paper is organised as follows. In Section 2 a summary of related work is presented. In Section 3 we describe the evolutionary events we consider, how we extract them and detail our visualisation approach. Finally, in Section 4, we present conclusions and potential future directions for our tool.

[1] https://github.com/hhromic/dynamic-community-vis
[2] https://uimr.insight-centre.org/dcv/

2 Related Work

Finding communities in static graphs is a well-known problem in the literature [7, 13]. In recent years researchers have been more focused on the dynamic aspects of communities motivated by modern real-world social media. For example, extensions to the popular clique percolation method for evolutionary networks has been proposed [12]. This extension involves applying community detection to joint graphs of adjacent time steps. A similar life cycle model was proposed in [16], where the dynamic community finding task is formulated as a graph colouring problem. Another example is in [1], where a community event identification approach is described based on a matching strategy across time steps. Besides social communities, the more general problem of identifying clusters in dynamic networks has been studied extensively, e.g. [3, 4, 9, 5].

The visualisation and exploration of such dynamic communities has been presented extensively in previous work [17]. For example, the transition between time steps can be visualised using straight links in timelines [14] or splines [15]. The former is one of the few examples with an explicit sorting strategy. Moreover, communities or vertices have been depicted using changing nodes [6, 11]. However, to the best of our knowledge, no other work has previously used event drops in timelines for representation.

3 Tracking and Visualising Dynamic Communities

We present our visualisation approach as an extension to the dynamic community tracking algorithm proposed by Greene et al. in [8]. In their work, the authors propose a generalisation for the task of dynamic community finding focused on the life cycle of user communities in dynamic networks. In this paper, we refer to it as TRACKER.

A dynamic network is represented as a set of time step graphs that provide snapshots of the nodes and edges of the network at successive intervals. The dynamic community finding task is then defined as identifying a set of k dynamic communities $\mathbb{D} = \{D_1, \ldots, D_k\}$ that are present in the network across one or more consecutive time steps with any desired granularity, e.g. every minute, hour or day. Furthermore, a *step community* is defined as a static community identified at a particular time step, representing a specific observation of a dynamic community at a given point in time. It is important to note here that the identification of such step communities is not restricted to any particular community detection method or algorithm. The definition of communities can be of any nature desired and is solely dependant on the chosen static community identification approach. The set of k_t step communities identified at each time t is denoted as $\mathbb{C}_t = \{C_{t,1}, \ldots, C_{t,k_t}\}$. Therefore, each dynamic community D_i can be represented by a *timeline* of its step communities, ordered by time.

We are interested in visualising the evolution of of this set of dynamic communities $\mathbb{D}$. For this we propose a two-fold approach. First, we compute the dynamic community timelines for sets of step communities using TRACKER from Greene et al. Then, we post-process these timelines and compute explicit community evolutionary events (e.g. splits, merges, contractions and expansions) and their visualisation meta-data, e.g. the source/target steps for a split event or the grow ratio for an expansion event.

We aim for an intuitive visual representation of all the available evolutionary information at a glance and therefore we propose to use a modified *event drops* display[3]. In this type of visualisation, time is represented in the horizontal axis and dynamic communities D_i are stacked in lanes, with their step communities $\mathbb{C}_t$ represented using circles (drops) at each discrete time step t horizontally. This display can be zoomed in or out along the horizontal axis (time), and drops fuse or separate when they become close or distant enough. Originally, the event drops display is only capable of single disconnected drops per event. Thus, we extend it to include connecting lines between drops that represent the continuation, splitting, merging, contraction and expansion evolutionary events. An example view of our proposed visualisation can be seen in Figure 2.

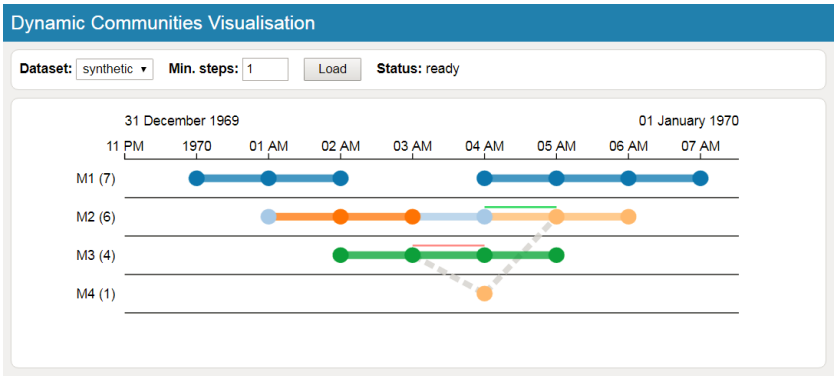


Fig. 2. Example Visualisation of Dynamic Timelines for Synthetic Communities.

In the Figure, four dynamic community timelines can be seen named from M_1 to M_4 that demonstrate all the considered evolutionary events. For each timeline M_i, static step communities exist with a particular attribute represented by a colour at different time steps. An example of an attribute can be a dominant topic label for each step communities. These attributes can change in time and therefore change colour. The timeline M_1 shows an intermittence. M_2 shows an expansion from the fourth to the fifth step, represented by a green line on top. M_3 shows both a contraction from step 2 to step 3, represented by a red line on top, and a split to M_4 in step 2, represented by a dashed line from M_3 to M_4. Lastly, M_4 shows a merge back to M_2 also represented by a dashed line between both. In all cases, the continuation lines adopt the same colour of the attribute for the next step community in the same dynamic community.

3.1 Extracting Evolutionary Events

In the dynamic community finding literature (e.g. [12, 16, 1]) there is a consensus on the fundamental events that can be used to characterise the evolution of dynamic communities. Using the same notation as in the previous Section, we consider the following evolutionary events for visualisation:

[3] https://github.com/marmelab/EventDrops

Birth is the emergence of a step community $C_{t,i}$ observed at time t for which there is no corresponding dynamic community $D_i \in \mathbb{D}$.

Death is the dissolution of a dynamic community D_i after not observing at least d corresponding consecutive step communities for it in $\mathbb{C}_t$. With d being user-defined.

Merging occurs when two different dynamic communities (D_i, D_j) observed at time $t-1$ share the same single step community at time t. Consequently, the pair starts sharing common timelines.

Splitting occurs when a single dynamic community D_i at time $t-1$ observes two distinct step communities at time t. A branching occurs and a new dynamic community D_j emerges at time t with a new independent timeline.

Contraction and Expansion are the change in size, i.e. number of members, of a dynamic community D_i from time $t-1$ to t, that is under or above a user-defined threshold α, e.g. $\alpha > 10\%$ for an expansion.

Intermittence occurs when a dynamic community D_i is not observed from time $t-1$ to t but re-appears in a future user-defined time $t+n$ before its death event.

Attribute Change is the change of a user-defined attribute for a dynamic community D_i, e.g. its topic, from time $t-1$ to t. This event was not previously considered by Greene et al. in their original work [8] and is an addition from ours.

The TRACKER algorithm computes the timelines for dynamic communities $\mathbb{D}$ from an input set of step communities $\mathbb{C}_t$, however it does not explicitly identify the above events in these timelines. Therefore, we present TRACKER2VIS, a post-processing algorithm that uncovers these evolutionary events from the plain TRACKER timelines.

To extract the evolutionary events, TRACKER2VIS uses both, the original input step communities and the output timelines generated by TRACKER, to compute a set of k' events $\mathbb{E} = \{E_1, \ldots, E_{k'}\}$ for the dynamic communities $\mathbb{D}$. An event E_i is composed of a $type$, e.g. *birth* or *expansion*, and depending on this type, a set of associated event meta-data: for *birth*, *death* and *intermittence*, the step community $C_{t,i}$ of the event; and for *merge*, *split*, *contraction*, *expansion* and *attribute change*, the source and target step communities $(C_{t,i}, C_{t+1,i})$ of the event. For the $contraction$ and $expansion$ types we also record the shrink and grow ratio $r > \alpha$, and for *attribute change* its new value.

To compute the set of events $\mathbb{E}$, we proceed as follows. From the step communities, we can trivially extract the *expansion* and *contraction* evolutionary events by comparing every adjacent step communities $C_{t,i} \in \mathbb{C}_t$ and $C_{t+1,i} \in \mathbb{C}_{t+1}$ of each dynamic community $D_i \in \mathbb{D}$, measuring their sizes against a predefined threshold α. Then we record the source and target step communities for which this size changed above α.

The rest of the evolutionary events are extracted from the timelines computed by TRACKER. First, we find *split* events by searching dynamic communities pair-wise $(D_i, D_j) \in \mathbb{D}$ for those with common first step community. Then for each found pair, we iterate their common step communities until we find a distinct step, thus signalling a divergence in their timelines. We record the source and target step communities where the divergence happens and proceed to remove all the initial step communities of the dynamic community D_j until this point. After this operation, it is possible that the timelines now contain duplicated dynamic communities, therefore we find and remove these duplicates. To find *merge* events, we follow a similar approach but in reverse order of step communities iteration, i.e. from back to front. At this point, it is now possible that some previously found *split* events are left orphaned, i.e. finding the *merge* events removed the step communities they referenced. Therefore we further scan the built *split*

events and remove those that are no longer relevant. Finally, we proceed to find the *birth*, *death* and *intermittence* events by iterating every single dynamic community $D_i \in \mathbb{D}$. For *birth* and *death*, we search for the first and last step communities respectively and record them as the event meta-data. For *intermittence*, we scan the timelines for non-consecutive step communities that are not the last and record this as the event.

3.2 Visualising Timelines and Evolutionary Events

After extracting a rich set of evolutionary events from dynamic community timelines, we can now visualise them using our proposed modified event drops display.

Summarising the example in Figure 2, the *expansion* and *extraction* events are visualised using green and red lines respectively between the two involved step communities $C_{i,t}$ and $C_{i,t+1}$. If the users of a static step community $C_{i,t}$ *split*, this is represented by a dashed line from their original community to a new dynamic community in a different lane. Likewise, if the members of a step community $C_{i,t}$ in a dynamic community D_i join another existing dynamic community D_j in a consecutive time step $C_{j,t+1}$, the *merge* is represented by a dashed line from $C_{i,t}$ to $C_{j,t+1}$. Lastly, every community can contain an arbitrary user-defined attribute whose values are represented by a colour mapping in both, the circle (drops) representation and the continuation lines. This provides a clear visualisation of *attribute change* in the dynamic communities.

We further illustrate our proposed tool using a real-world use-case. In the context of online social awareness and recommendation, in previous work [2] we captured Twitter data related to TV programmes being broadcasted live in Ireland by the national television broadcaster, RTÉ. We then constructed sets of step communities $\mathbb{C}_t$ discovered using the OSLOM algorithm [10] which is based on the modularity of the groups. We applied the TRACKER algorithm and our TRACKER2VIS post-processing to obtain a set of dynamic community timelines $\mathbb{D}$ and their set of evolutionary events $\mathbb{E}$. For the user-defined community attribute, we used a topic label defined as the name of the TV programme most frequently referenced by the community members. The extraction of these topic labels is not restricted to any particular method. For example, topics could also be defined from the top three most frequent hashtags in the community. Figure 3 shows an example visualisation of the DCV tool applied to this data, where only dynamic communities with at least three step communities are displayed for clarity.

The DCV shows that some dynamic communities, such as M_3, M_{27} and M_{29} are steady and stable in time, while others such as M_{12}, M_{14}, M_{21} and M_{26} result in quite intermittent dynamic communities. After manual examination of these cases, we discovered that they were very short-lived communities, as identified by the OSLOM algorithm, however they tend to re-emerge later as new dynamic communities. This observation is recurrent in other dynamic communities in the dataset and is not a desired behaviour since it can be misleading for a decision maker, who for example could conclude that these short-lived communities are not performing well, while in fact they are just not being detected as the same community. While the OSLOM algorithm is reportedly a good community detection method for classic social media datasets [10], for the case of Twitter data it seems to be less effective.

Overall, examination using our visualisation tool then suggests to the analyst that an alternative detection method might be necessary. In particular, one that is more robust to the highly dynamic nature of the microblogging platform.

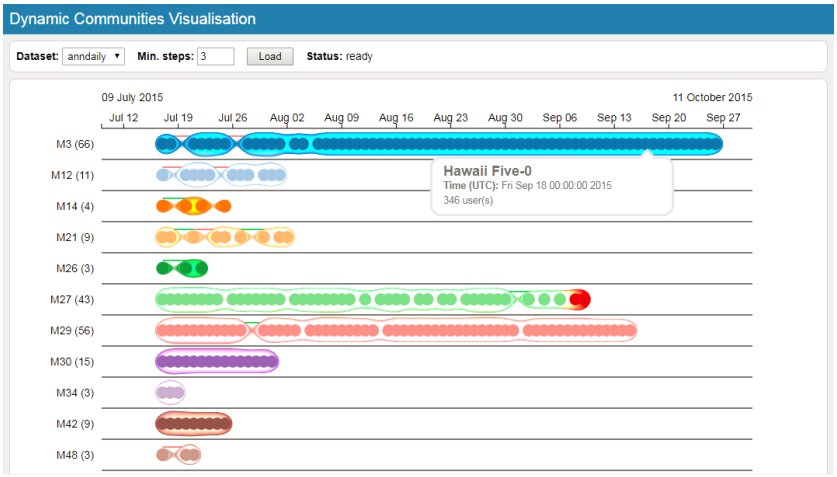


Fig. 3. The DCV tool applied to real-world RTÉ Twitter data.

4 Conclusions

In this work we presented the *Dynamic Community Viewer* (DCV), a tool for visualisation of dynamic community timelines and their evolutionary events. To achieve this, we (1) built a post-processing algorithm, TRACKER2VIS, on top of the dynamic communities tracking algorithm TRACKER [8], for extracting common evolutionary events from the literature such as *birth*, *death*, *split*, *merge*, *contraction*, *expansion*, and (2) proposed an additional event type *attribute change* to further support the visualisation of user-defined aspects of the communities such as their topic. We believe that this visualisation tool is a valuable resource for researchers, decision makers and professionals in the field of dynamic networks and community analysis. We also showcased an example use-case from our past work in the Twitter microblogging scenario.

For future work, we consider two main lines for improvement: (1) a method for sorting and/or grouping the dynamic timelines, e.g. one could wish to see dynamic communities that split/merge more often together or dynamic communities with similar topics sorted by usage frequency, and (2) real-time updating of the visualisation. While the latter is an ambitious goal, it can be interesting to adapt our visualisation for automatically refreshing according to live step communities being identified on the fly.

References

[1] Sitaram Asur, Srinivasan Parthasarathy, et al. “An Event-based Framework for Characterizing the Evolutionary Behavior of Interaction Graphs”. In: *ACM Trans. Knowl. Discov. Data* 3.4 (Dec. 2009).

[2] Andrea Barraza-Urbina, Hugo Hromic, et al. “Using Social Media Data for Online Television Recommendation Services at RTÉ Ireland”. In: *ACM Recommendation Systems Conference*. 2nd Workshop on Recommendation Systems for Television and Online Video. Sept. 2015.

[3] Deepayan Chakrabarti, Ravi Kumar, et al. “Evolutionary Clustering”. In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’06. New York, NY, USA: ACM, 2006.

[4] Yun Chi, Xiaodan Song, et al. “Evolutionary Spectral Clustering by Incorporating Temporal Smoothness”. In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’07. New York, NY, USA: ACM, 2007.

[5] Evgenia Dimitriadou, Andreas Weingessel, et al. “A Combination Scheme for Fuzzy Clustering”. In: *Advances in Soft Computing — AFSS 2002*. AFSS International Conference on Fuzzy Systems. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, Feb. 3, 2002.

[6] Tanja Falkowski, Jorg Bartelheimer, et al. “Mining and Visualizing the Evolution of Subgroups in Social Networks”. In: *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*. WI ’06. Washington, DC, USA: IEEE Computer Society, 2006.

[7] Santo Fortunato. “Community detection in graphs”. In: *Physics Reports* 486.3 (Feb. 2010).

[8] D. Greene, D. Doyle, et al. “Tracking the Evolution of Communities in Dynamic Social Networks”. In: *2010 International Conference on Advances in Social Networks Analysis and Mining*. 2010 International Conference on Advances in Social Networks Analysis and Mining. Aug. 2010.

[9] Derek Greene and Pádraig Cunningham. “Multi-view clustering for mining heterogeneous social network data”. In: *Workshop on Information Retrieval over Social Networks*. 31st European Conference on Information Retrieval (ECIR’09). Mar. 2009.

[10] Andrea Lancichinetti, Filippo Radicchi, et al. “Finding Statistically Significant Communities in Networks”. In: *PLoS ONE* 6.4 (Apr. 29, 2011).

[11] M. Ogawa, K. l Ma, et al. “Visualizing social interaction in open source software projects”. In: *2007 6th International Asia-Pacific Symposium on Visualization*. 2007 6th International Asia-Pacific Symposium on Visualization. Feb. 2007.

[12] Gergely Palla, Albert-László Barabási, et al. “Quantifying social group evolution”. In: *Nature* 446.7136 (Apr. 2007).

[13] Symeon Papadopoulos, Yiannis Kompatsiaris, et al. “Community detection in Social Media”. In: *Data Mining and Knowledge Discovery* 24.3 (June 14, 2011).

[14] Khairi Reda, Chayant Tantipathananandh, et al. “Visualizing the Evolution of Community Structures in Dynamic Social Networks”. In: *Computer Graphics Forum* 30.3 (2011).

[15] Martin Rosvall and Carl T. Bergstrom. “Mapping Change in Large Networks”. In: *PLOS ONE* 5.1 (2010).

[16] Chayant Tantipathananandh, Tanya Berger-Wolf, et al. “A Framework for Community Identification in Dynamic Social Networks”. In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’07. New York, NY, USA: ACM, 2007.

[17] Corinna Vehlow, Fabian Beck, et al. “Visualizing Group Structures in Graphs: A Survey”. In: *Computer Graphics Forum* 36.6 (2017).

A Case-Study on the Impact of Dynamic Time Warping in Time Series Regression

Vivek Mahato and Pádraig Cunningham

School of Computer Science
University College Dublin
Dublin 4, Ireland
vivek.mahato@ucdconnect.ie, padraig.cunningham@ucd.ie

Abstract. It is well understood that Dynamic Time Warping (DTW) is effective in revealing similarities between time series that do not align perfectly. In this paper, we illustrate this on spectroscopy time-series data. We show that DTW is effective in improving accuracy on a regression task when only a single wavelength is considered. When combined with k-Nearest Neighbour, DTW has the added advantage that it can reveal similarities and differences between samples at the level of the time-series. However, in the problem, we consider here data is available across a spectrum of wavelengths. If aggregate statistics (means, variances) are used across many wavelengths the benefits of DTW are no longer apparent. We present this as another example of a situation where big data trumps sophisticated models in Machine Learning.

Keywords: Regression, Time Series Data, Dynamic Time Warping.

1 Introduction

In this paper, we consider the task of predicting the etch rate in silicon wafer production. While we frame this as a prediction task, our interest is in gaining an insight into the sources of process variability in plasma etching rather than very accurate prediction. Because we have insight as an objective we are interested in k-Nearest Neighbour (k-NN) classifiers because of their interpretability.

In the production of silicon wafers, the etch rate is critically important because it directly influences the precision of the silicon structures. If a transistor gate length is too large the device speed is reduced, it if is too small it will leak. So it is important to be able to control the etch rate very precisely[7, 2].

It is common to describe plasma etching as a cooking process or a recipe. As the recipe progresses different gases are introduced in the chamber to promote specific effects. Optical emission spectroscopy (OES) is the most widely used technique to track what is going on within the chamber [9].

In the data we analyze here, the outcome is the etch rate, and the predictive variables are time-series spectra across 2048 wavelengths. The process takes about 42 seconds, the spectra are sampled at 1.3Hz so each of the 2048 OES time-series contains about 57 ticks [7]. With our objective of identifying sources

of process variability, we are interested in identifying systematic differences in OES time-series that are predictive of differences in etch rate.

In the next section, we introduce the problem domain and describe the data on which the analysis is based. In section 3 we describe the prediction models we use and we discuss the results of the evaluation in section 4. Some conclusions and directions for future work are presented in section 5.

2 The Problem Domain

Plasma etching is an integral part of the pipeline in wafer manufacturing. This process is conducted within specialised etch chambers where etchant gases in plasma form are directed towards the wafer surface where the exposed material is removed. This removal of material contributes to the formation of semiconductor components on the wafer [2, 6, 5].

The etching process involves chemical reactions that emit a spectrum of frequencies of electromagnetic radiation, which can be captured by a spectroscope. The captured spectroscopic data for each wafer is a high dimensional multivariate time series where each dimension represents one wavelength through a period of time during the etching process [7].

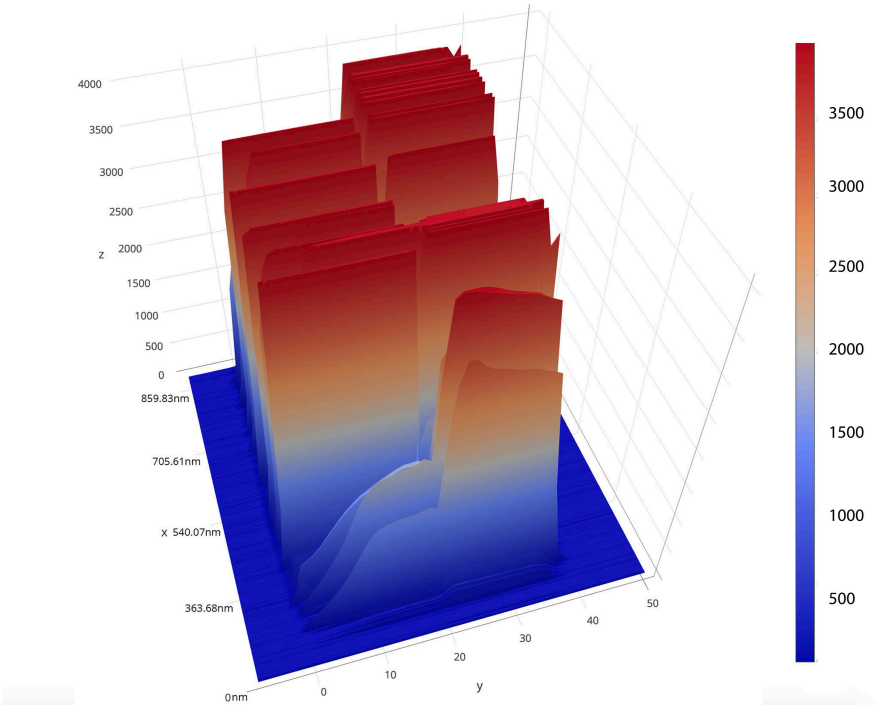


Fig. 1: Spectroscopic representation of a wafer datum.

Figure 1 shows the spectroscopic data for a typical plasma etching process in 3D. There are 2048 wavelengths plotted on the x-axis but it is clear that interesting peaks show up only on a small subset of these. Time is plotted on the y-axis and the height (z-axis) shows the intensity.

In the analysis presented here the machine learning task is to predict the etch rate from this time-series data (a regression task). In earlier work, this has been done by representing the time series by four summary statistics and using multivariate regression on these statistics [6, 5]. The four statistics are produced by dividing the time series into two stages and representing each stage using mean and standard deviation. The two stages are clearly visible in Figure 1.

2.1 The Data

Altogether we have data on the production of 1771 wafers. For each of these wafers, we have an estimate of the average etch rate which we take as the dependent variable. As stated already, our ML objective is to predict the etch rate, a regression task. For each wafer, we have the full OES time-series as shown in Figure 1. This is sampled across 2048 wavelengths and contains about 57 ticks for each wavelength.

Rather than consider all 2048 wavelengths, in consultation with a domain expert, we reduced this to 38 wavelengths with peaks known to correspond to the key compounds involved in the plasma etching process.

It is clear from looking at the spectra in Figures 1 that the process has two distinct stages. For this reason, the time-series data can be significantly reduced by simply representing each spectrum time series by four aggregate statistics, the mean and standard deviation in stages 1 and 2. In this way, each wafer can be represented by 2048×4 data points rather than 2048×60 data points in the full time-series representation.

3 Prediction Models

Given our objective of gaining insights from the data we are interested in k-Nearest Neighbour regression using DTW on the spectra time-series as the similarity measure. The analysis of time-series classification algorithms presented by Bagnall *et al.* [1] shows that 1-NN DTW is an effective method for time-series classification, if not as accurate as state-of-the-art ensemble methods such as COTE.

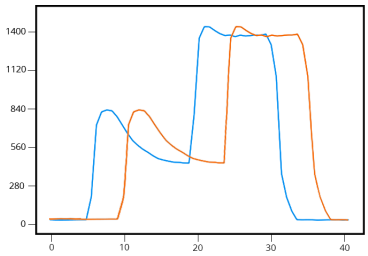
We compare k-NN DTW against two base-line predictors, multivariate regression on the aggregate statistics and k-NN on the aggregate statistics. So we consider three prediction models:

1. **R-4M** Multivariate regression using the four aggregate statistics.
2. **k-NN-4M** k-NN regression ($k = 5$) using the four aggregate statistics.
3. **k-NN DTW** k-NN using DTW ($k = 5$) as the similarity measure (see section 3.1 for details).

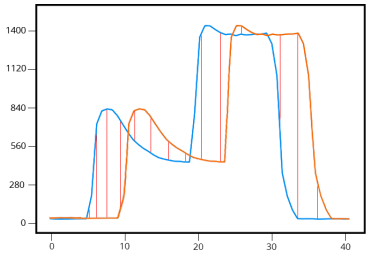
When these are compared they are always compared on the same underlying wavelengths; for instance in Figure 4 they are compared over 1 to 11 wavelengths.

3.1 Dynamic Time Warping

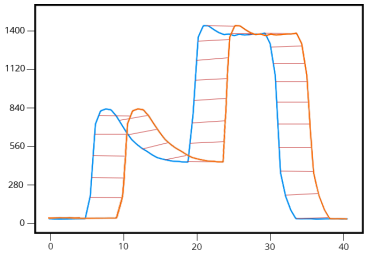
To find the distance between two data series, the Euclidean formula is a popular choice. But when dealing with time-series data where the series may be displaced in time, the Euclidean distance may be large when the two series are similar, just off slightly on the timeline. To tackle this situation Dynamic Time Warping offers us the malleability of mapping the two data series in a non-linear fashion by warping the time axis [4]. It creates a cost matrix where the cells contain the distance value of the corresponding data-points and then finds the shortest path through the grid, which minimizes the total distance between them.



(a) Two time series displaced in time.



(b) Mapping of data-points without warping.



(c) Mapping of data-points with warping.

Fig. 2: Dynamic Time Warping.

The figure above illustrates the non-linear mapping done by DTW when we give it a suitable extent of data-points, i.e. max warping window, to consider while fetching the shortest path. Initially, when we restrict our algorithm with no warping being allowed, the data points are linearly mapped between the two data series based on the common time axis value. As seen in Figure 2(b), the algorithm fails to map the trend of a time series data to another. But when we grant the DTW algorithm the flexibility of considering a warping window,

the algorithm performs remarkably when mapping the data-points following the trend of the time-series data, which can be visualized in Figure 2(c).

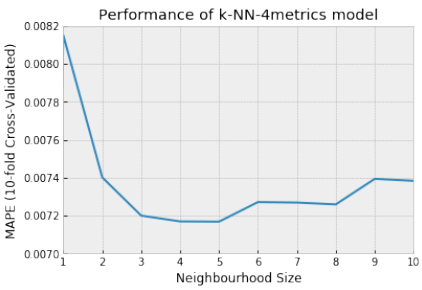
4 Evaluation

As mentioned in the Introduction, we are more concerned with gaining insights into the factors that influence outcomes that the ability to predict the outcomes themselves. For this reason, we are evaluating the potential of DTW to reveal systematic patterns in the data.

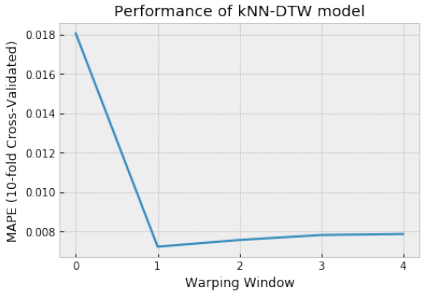
4.1 Parameter Selection

As stated in section 2.1, domain knowledge indicates that the 2048 wavelengths can be reduced to 38. Through a preliminary feature selection analysis, this was reduced further to 11 wavelengths that were found to be predictive for at least one of the three models. In the next subsection we report on the impact of these 11 wavelengths on prediction accuracy then we conclude the evaluations with some comments on insights.

For k-NN neighbourhood size (k) is a crucial parameter governing performance. Fig. 3(a) shows the impact of different values of k for the k-NN-4M model. Given that the best performance is achieved when $k = 5$ we use this for both k-NN models.



(a) k-NN-4M performance when trained on 4 summary metrics of a single wavelength against different neighbourhood sizes.



(b) k-NN-DTW performance when trained on a single wavelength against different warping-window sizes.

Fig. 3: Effect on the performance of k-Nearest Neighbour models with different parameters.

For DTW the size of the warping window has an impact [8] so we tried different values between 0 and 4. Fig. 3(b) shows the impact of the warping-window on the performance of k-NN-DTW model trained on a single wavelength. The

model performs the worst when we set the warping-window size to be 0, which is a simple Euclidean distance calculation between the two time-series data. The model performs significantly better when we allow warping but the performance does not improve above a warping window size of one so that parameter value was chosen.

4.2 Prediction Accuracy

Figure 4 summarizes the results of a 10-fold cross-validation analysis of the three prediction models using the 11 wavelengths. For each wavelength, the k-NN-DTW model has access to the full time-series representation and the other two models use the four summary statistics only.

Unsurprisingly, when only a single wavelength is considered, the Mean Absolute Percentage Error (MAPE) of the DTW model is significantly better than the models using the four summary statistics. However, as more wavelengths are added the performance of the k-NN-DTW model does not improve and the other models close the gap. If prediction accuracy were our core objective then it seems

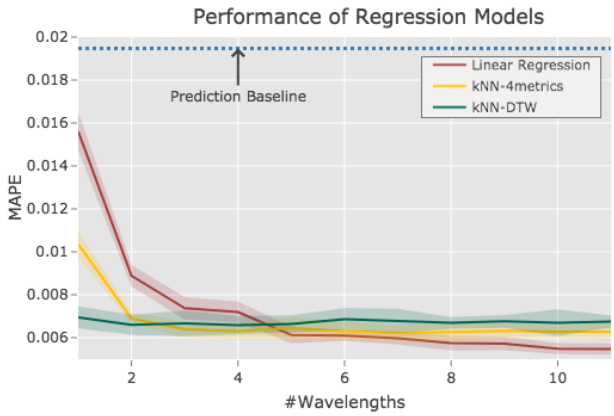


Fig. 4: Comparison between different regression models. The shading shows the standard deviation.

that DTW does not offer any benefits because a feature vector based on summary statistics from a wider set of wavelengths provides accuracy that is even slightly better. The results in Figure 4 suggest that k-NN with four summary statistics from 11 wavelengths (44 features) has the lowest MAPE score.

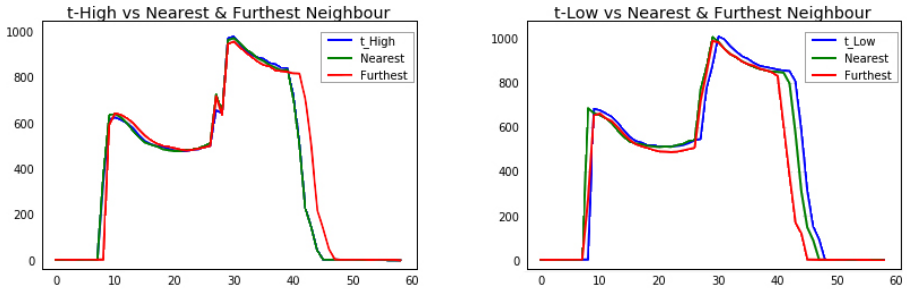
This might be interpreted as a variation on the modern wisdom that "*more data is better than smarter algorithms*" [3]. It is a variation in the sense that the benefits of big data are understood to derive from a large number of examples. Whereas in our case the advantage comes from more extensive data on each

example. If summary statistics are available across many wavelengths there is no benefit in using DTW (the smart algorithm in this case).

4.3 Insights

The objective in using DTW to identify nearest neighbours was to see if an interpretable measure of similarity would offer insights into the factors influencing outcomes. To test this we selected two wafers, one with the highest etch rate and one with the lowest etch rate and examined their nearest and furthest neighbours as measured by DTW on the first wavelength in Figure 4 (564.96nm).

Looking at the wafer with the high etch rate first (Figure 5 (a)) we see that it matches its nearest and furthest neighbours well all along the time series until the end of the second phase where the intensity of the wavelength starts to drop off. It is clear that the nearest neighbour matches perfectly here while this region of the time series accounts for the main difference between the query case and it's most distant neighbour.



(a) Wafer with highest etch rate.

(b) Wafer with lowest etch rate.

Fig. 5: Nearest and furthest neighbours of a wafer based on a single informative wavelength.

When we turn to the wafer with the low etch rate (Figure 5 (b)) it appears that the same region of the time-series seems also to account for the difference between the query case and the most different neighbour. This supports the conclusion that this phase of the process is the source of variability in the etch rate.

5 Conclusions & Future Work

While the work reported here is about predictive analytics the motivation is more to discover insights into factors that influence outcomes rather than generating very accurate predictions. For this reason, we focus on k-NN DTW as a model that is interpretable.

While this research is still ongoing we report two findings here:

- Using DTW does improve prediction accuracy over models that use summary statistics of the time series on a single wavelength. However, when other wavelengths are considered the benefits of using DTW are no longer apparent.
- DTW does seem to offer interpretability benefits through the analysis of nearest and furthest neighbours to identify regions of the time series where differences are apparent.

6 Acknowledgments

This publication has resulted from research supported in part by a grant from Science Foundation Ireland (SFI) under Grant Number 16/RC/3872 and is co-funded under the European Regional Development Fund.

References

1. Bagnall, A., Lines, J., Bostrom, A., Large, J., Keogh, E.: The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery 31(3), 606–660 (2017)
2. Flamm, D.L., Donnelly, V.M.: The design of plasma etchants. Plasma Chemistry and Plasma Processing 1(4), 317–363 (1981)
3. Halevy, A., Norvig, P., Pereira, F.: The unreasonable effectiveness of data. IEEE Intelligent Systems 24(2), 8–12 (2009)
4. Keogh, E.J., Pazzani, M.J.: Derivative dynamic time warping. In: Proceedings of the 2001 SIAM International Conference on Data Mining. pp. 1–11. SIAM (2001)
5. Lynn, S.A., MacGearailt, N., Ringwood, J.V.: Real-time virtual metrology and control for plasma etch. Journal of Process Control 22(4), 666–676 (2012)
6. Lynn, S.A., Ringwood, J., MacGearailt, N.: Global and local virtual metrology models for a plasma etch process. IEEE Transactions on Semiconductor Manufacturing 25(1), 94–103 (2012)
7. MacGearailt, N.: Process diagnostics of industrial plasma systems. Ph.D. thesis, Dublin City University (2015)
8. Ratanamahatana, C.A., Keogh, E.: Three myths about dynamic time warping data mining. Proceedings of the 2005 SIAM International Conference on Data Mining p. 506510 (2005)
9. Shul, R., McClellan, G., Briggs, R., Rieger, D., Pearton, S., Abernathy, C., Lee, J., Constantine, C., Barratt, C.: High-density plasma etching of compound semiconductors. Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films 15(3), 633–637 (1997)

Multiple-Kernel Dictionary Learning for Sparse Reconstruction of Unseen Multivariate Time-series *

Babak Hosseini[0000−0003−3335−4289] and Barbara Hammer

CITEC center of excellence, Bielefeld University, Bielefeld, Germany
{bhosseini,bhammer}@techfak.uni-bielefeld.de

Abstract. A considerable amount of research has been devoted to designing algorithms for description and recognition of unseen classes in datasets. One important application of such algorithms is multivariate time-series (MTS) such as motion data and audio signals, in which it is usually expected to observe partial similarities between specific dimensions of the unseen and seen classes. In this work, we propose a novel unsupervised framework for the analysis of the unseen categories in MTS datasets from the above perspective. Our framework learns a unique multiple-kernel dictionary (MKD) which selects and combines specific dimensions of the MTS taken from the seen classes. We incorporate the designed MKD in a non-negative sparse coding framework (MKD-SC) to obtain an encoding that fully/partially describes the unseen MTS categories. This achieved description is interpretable via finding possible relations between the data dimensions of the seen and unseen classes. Furthermore, by treating such dimension-based sparse encodings as compact semantic attributes, we propose a simple but effective incremental clustering algorithm to categorize the unseen data classes in an unsupervised way based on those encoded attributes. Our incremental clustering can gradually build a hierarchal tree via on-line observation of the unknown (unseen) MTS samples. Empirical evaluation on real benchmarks demonstrates the effectiveness of our MKD-SC framework in the interpretable reconstruction of the unseen data regarding their dimension-based overlapping with the seen classes. It also shows the effect of the learned attributes (encodings) on improving the clustering performance of the unseen MTS categories.

Keywords: Multivariate time-series, dictionary learning, unseen classes

1 Introduction

Zero-shot learning is the problem of recognizing novel categories of data when no prior information is available during the training phase [2, 9, 15]. One effective approach to such transfer learning is the incorporation of semantic attributes as descriptive features to map the input data to an intermediate semantic space, which can discriminate between different unseen categories [9, 15]. Another concern in this area of research is the partial/complete reconstruction of the unseen classes based on their relation to the learned semantic attributes or to the training data [12, 13].

One important application of zero-shot learning is multivariate time-series (MTS) in the general meaning such as audio data and human motions [4, 11] with a considerable

* This research was supported by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).

number of unknown classes. Different from image and video data, MTS do not possess any general spatial dependency between its dimensions. Nevertheless, it is usually expected to find semantic attributes shared between different classes of a MTS dataset. As an example of MTS data, consider the Cricket Umpire signal *Out* in Figure 1, which can be described as the *left hand is raised* while *the right hand is down*. This can provide us with a semantic understanding of the data without having any prior knowledge about its class. Furthermore, such descriptions can be considered as semantic attributes in order to distinguish the unknown MTS data samples into distinct categories which reflect their unknown ground truth labels. Although the semantic descriptions are class specific, the individual attributes could be shared among different classes with partial similarities.

Sparse coding (SRC) is the idea of constructing an input data using weighted combinations (*sparse codes*) of sparse selected entries from a set of learned bases (*dictionary*). Such sparse representations can capture essential intrinsic characteristics of a dataset [14]. Furthermore, via assuming an implicit mapping of the data to a high-dimensional feature space, it is possible to formulate SRC using the kernel representation of the data [8] to also model nonlinear data structures. Consequently, a subset of the existing research has benefited from SRC methods in designing more effective attributes for dealing with unseen classes of data; however, these efforts are mainly limited to the image (spatial) and video (spatiotemporal) datasets [13, 16]

Despite the current achievements in learning unseen MTS data, the existing methods are either depended on having prior information about the novel classes (e.g samples/labels) [11], or they cannot interpret the unseen data based on their learned attributes. Furthermore, to our knowledge, there is no research reported on the partial/complete reconstruction of unseen classes for MTS data in general (e.g. recorded motion signals).
Contributions: In this paper, We propose a novel framework for having a semantic description and categorizing the unseen classes in a MTS dataset based on dictionary learning in the feature space. The following contributions distinguish our work from the relevant literature:

– We design a novel dictionary structure which learns attributes that are effective in representing multivariate time-series based on their individual dimensions.

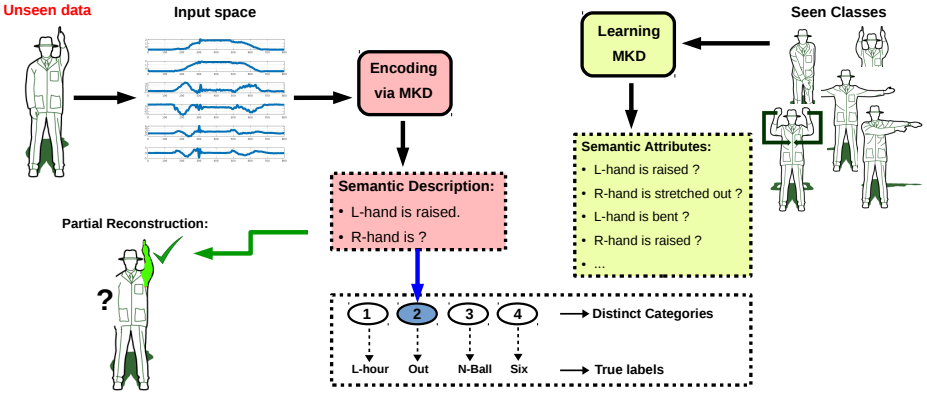


Fig. 1: General overview of our framework. The dictionary (MKD) learns the semantic attributes based on the seen classes. These attributes are used for a semantic description of the data from the unseen classes, which leads to categorizing and partial reconstruction of the data.

– We propose an unsupervised kernel-based SRC method for partial reconstruction of unseen MTS data in the feature space along with their interpretable encoding.
– We design an incremental clustering algorithm based on the sparse encodings of the unseen data which gradually creates a clustering dendrogram of the unseen classes.

In the next section, we introduce and explain our proposed framework, and we evaluate it in Sec. 4, followed by the conclusion section.

2 Problem Statement

Presenting a multivariate time-series in the vectorial space, each data sequence i is denoted by $\mathbf{X}_i = [\boldsymbol{x}_i(1) \dots \boldsymbol{x}_i(T)] \in \mathbb{R}^{f \times *}$, where $(f, *)$ represents the number of dimensions and different lengths for (T) respectively in the MTS sequences. Assuming an implicit data tensor $\mathcal{X} = \{\mathbf{X}_i\}_{i=1}^{N} \in \mathbb{R}^{N \times f \times *}$ contains the observed data classes, its corresponding label vector $\boldsymbol{l} \in \mathbb{R}^{N}$ maps $\mathcal{X}$ to c distinct known data classes. On the other hand, there exists a set of unseen MTS data $\mathcal{Z} \in \mathbb{R}^{N_z \times f \times *}$ with a corresponding label vector $\boldsymbol{q} \in \mathbb{R}^{N_z}$ such that $\boldsymbol{q} \cap \boldsymbol{l} = \emptyset$. Based on the above description, we are interested in solving the following problems:

– Obtaining semantic descriptions (Figure 1) for each sequence $\mathbf{Z}_i \in \mathcal{Z}$ which can explain/reconstruct the dimensions of $\mathbf{Z}_i$ based on the seen classes $\mathcal{X}$.
– Using the achieved semantic descriptions to distinguish the sequences of $\mathcal{Z}$ into distinct categories which correspond to the unknown label vector $\boldsymbol{q}$.

3 Multiple-Kernel Dictionary Learning Framework

Similar to Figure 1, it is often expected in the real-life problems (e.g. human motions) to find partial similarities between the seen and unseen classes of MTS data for a subset of their dimensions. Therefore, these similarities can lead to an interpretable description of the unknown classes. Such a description can also lead to better categorizing of the novel data without having any prior information on its class labels. To achieve the above, we design a specific multiple-kernel dictionary (MKD) structure which is trained based on $\mathcal{X}$ and learns semantic attributes similar to Figure 1. This is done via combining dimensions of similar MTS samples in the feature space under non-negativity constraints. These attributes can encode each unseen $\mathbf{Z}_i \in \mathcal{Z}$ leading to an interpretable description of its dimensions and to better separate it from previous (unknown) classes in $\mathcal{Z}$ (Figure 1).

To be more specific, we assume there exists f individual non-linear implicit functions $\{\Phi_i(\mathbf{X})\}_{i=1}^{f}$ for mapping the respective dimensions of the sequence $\mathbf{X}$ individually into RKH-spaces corresponding to kernels $\mathcal{K}_i$ [8]. A weighted combination of these kernels with coefficients $\beta_i \geq 0$ induces an embedding of the data $\Phi(\mathbf{X}, \boldsymbol{\beta}) := [\sqrt{\beta_1}\Phi_1(\mathbf{X})^\top \cdots \sqrt{\beta_f}\Phi_f(\mathbf{X})^\top]^\top$ in the feature space, which can be obtained via the vector $\boldsymbol{\beta} \in \mathbb{R}^f$ and applies different weights to the individual dimensions.

We assume that cluster structures are complemented by different weighting schemes of the individual kernels. Thus, we consider k different weighting schemes and denote $\mathbf{B} = [\boldsymbol{\beta}_1 \cdots \boldsymbol{\beta}_k] \in \mathbb{R}^{f \times k}$ as the weighting matrix. Kernel sparse coding selects dictionary functions as weighted sums of elements of this embedding. We define the novel multiple kernel dictionary (MKD) matrix $\Phi_{\mathbf{B}}(\mathbf{U})$ as

$$\Phi_{\mathbf{B}}(\mathbf{U}) := [\Phi(\mathcal{X}, \boldsymbol{\beta}_1)\boldsymbol{u}_1 \cdots \Phi(\mathcal{X}, \boldsymbol{\beta}_k)\boldsymbol{u}_k] \qquad \text{where } \mathbf{U} = [\boldsymbol{u}_1 \dots \boldsymbol{u}_k] \in \mathbb{R}^{N \times k} \quad (1)$$

Each dictionary column $\Phi(\mathcal{X}, \boldsymbol{\beta}_i)\boldsymbol{u}_i$ is a weighted combination of dimensions and samples from $\mathcal{X}$ which can represent semantic attributes similar to those of Figure 1. The coefficient matrix $\mathbf{U}$ of the dictionary is subject of an optimization, to obtain the best representational power for the given data.

Sparse coding aims for a reconstruction of the data $\mathcal{X}$ in the kernel space via the MKD: $\Phi(\mathcal{X}) \approx \Phi_{\mathbf{B}}(\mathbf{U})\mathbf{\Gamma}$ with a matrix of sparse codes $\mathbf{\Gamma} = [\boldsymbol{\gamma}_1 \dots \boldsymbol{\gamma}_N] \in \mathbb{R}^{k \times N}$. We propose the following MKD sparse coding framework (MKD-SC) for training the dictionary parameters $(\mathbf{B}, \mathbf{U})$ and sparse codes $\mathbf{\Gamma}$:

$$\begin{aligned} &\min_{\mathbf{B},\mathbf{\Gamma},\mathbf{U}} \|\Phi(\mathcal{X}) - \Phi_{\mathbf{B}}(\mathbf{U})\mathbf{\Gamma}\|_F^2 + \tfrac{\lambda}{2}\sum_{i=1}^{k}\sum_{s,t=1}^{N} u_{is}u_{it}\|\Phi(\mathbf{X}_s, \boldsymbol{\beta}_i) - \Phi(\mathbf{X}_t, \boldsymbol{\beta}_i)\|_2^2 \\ &\text{s.t.} \quad \|\boldsymbol{\gamma}_i\|_0 < T_0, \;\; \|\Phi(\mathcal{X}, \boldsymbol{\beta}_i)\boldsymbol{u}_i\|_2^2 = 1, \;\; u_{ij}, \beta_{ij}, \gamma_{ij} \in \mathbb{R}^+, \;\; \forall ij \end{aligned} \tag{2}$$

where $\mathbf{\Gamma}$ encodes $\mathcal{X}$ via the sparse combinations of dimension-based attributes in $\Phi_{\mathbf{B}}(\mathbf{U})$. The scalar λ is the trade-off constant between different parts of the objective function, and $u_{ij}, \beta_{ij},$ and γ_{ij} address the i-th entry of the j-th column of $\mathbf{U}, \mathbf{B},$ and $\mathbf{\Gamma}$ respectively.

The rationale behind this objective is as follows: The first part of the loss term measures the reconstruction error of the sparse coding based on the Frobenius norm $\|.\|_F$. The term $\|.\|_0$ denotes the l_0-norm which employs sparsity constraints for elements of $\mathbf{\Gamma}$ via the constant T_0 which results in having each $\mathbf{X}_i$ constructed with sparse contributions from $\mathcal{X}$. The l_2-norm constraint on $\Phi(\mathcal{X}, \boldsymbol{\beta}_i)\boldsymbol{u}_i$ prevents the optimization solutions from becoming degenerated [14].

Inspired from [5], the second loss term in the objective of Eq. 2 enforces each dictionary vector $\boldsymbol{u}_i$ to use data samples from $\mathcal{X}$, such that their weighted features $\Phi(\boldsymbol{x}_s, \boldsymbol{\beta}_i)$ are local neighbors in the feature space. Hence this objective has a tendency to focus on dictionary elements which group similar time series together, and which chooses weighting scheme of the dimensions which enhance these similarities, i.e. cluster structures are mirrored in the dictionary elements and coefficient weighting schemes. Hence the dictionary $\Phi_{\mathbf{B}}(\mathbf{U})$, which results from the optimization problem in Eq. 2, contains attributes, which are weighted combinations of different exemplars and dimensions from $\mathcal{X}$. The non-negativity constraints result in having similar resources combined together which leads to learning semantic attributes for $\Phi_{\mathbf{B}}(\mathbf{U})$ and an interpretable sparse description based on each $\boldsymbol{\gamma}_i$ [7].

Estimating the Sparse Codes of Unseen data: Encoding an unseen data $\boldsymbol{z}$ based on the trained $\{\mathbf{U}, \mathbf{B}\}$ is possible via optimizing the reconstruction error of this problem, i.e. optimizing Eq. 2 with $\lambda = 0$. In the Sec. 3.2 and 3.3, we benefit from this framework to describe and categorize unseen multivariate time-series samples.

3.1 Optimization Scheme

We optimize the parameters $\mathbf{U}$, $\mathbf{\Gamma}$ and $\mathbf{B}$ in alternating steps, such that at each update step, we optimize Eq. 2 with respect to one parameter while fixing the others. Similar to [5], via using the associated kernel matrices $\{\mathcal{K}_i(\mathcal{X}, \mathcal{X}) = \Phi_i(\mathcal{X})^\top \Phi_i(\mathcal{X})\}_{i=1}^{f}$ it is possible to formulate Eq. 2 in terms of each $\boldsymbol{\gamma}_i$, $\boldsymbol{u}_i$, and $\boldsymbol{\beta}_i$ individually and without an explicit reference to the embeddings Φ_i, which leads to 3 constrained non-negative quadratic programmings with the general form of

$$\min_{\boldsymbol{x}} \tfrac{1}{2}\boldsymbol{x}^\top \mathbf{H}\boldsymbol{x} + \boldsymbol{c}^\top \boldsymbol{x} \qquad \text{s.t. } \|\boldsymbol{x}\|_0 < T_0, \;\; x_i \in \mathbb{R}^+ \;\; \forall i \tag{3}$$

These problems can be optimized via the non-negative quadratic pursuit (NQP) algorithm from [6]. Thus, We take the following steps for each iteration of the training phase:

1- Updating $\boldsymbol{\gamma}_i \ \ \forall i = 1, \ldots, N$
2- Updating $\boldsymbol{u}_i$ and $\boldsymbol{\beta}_i$ in a subsequent order $\forall i = 1, \ldots, k$ Due to the page limit, we put the detail regarding the reformulation of Eq. 2 and the optimization steps in the online extended version of the paper [1].

The optimization framework of Eq. 2 is non-convex when considering $\{\mathbf{U}, \mathbf{\Gamma}, \mathbf{B}\}$ together. However, each of the 3 sub-problems related to the update of $\boldsymbol{\gamma}_i$, $\boldsymbol{u}_i$, so $\boldsymbol{\beta}_i$ are convex, and the main optimization loop converges in a limited number of steps.

3.2 Partial Reconstruction of Unseen Time-series

In realistic MTS datasets such as human actions, it is expected to observe partial similarities between the dimensions of different classes. Therefore, we define the following error measure for reconstruction of a selected set of dimensions $\mathcal{S}$ related to $\mathbf{Z}$:

$$\mathcal{J}^{\mathcal{S}}_{rec}(\mathbf{Z}, \mathbf{B}, \mathbf{U}) = \|\mathbf{I}^{\mathcal{S}}\Phi(\mathbf{Z}) - \mathbf{I}^{\mathcal{S}}\Phi_{\mathbf{B}}(\mathbf{U})\mathbf{\Gamma}\|_2^2 / \|\mathbf{I}^{\mathcal{S}}\Phi(\mathbf{Z})\|_2^2 \tag{4}$$

where $\mathbf{B}^{\mathcal{S}}$, and $\mathbf{I}^{\mathcal{S}}$ are modified versions of $\mathbf{B}$ and the identity matrix respectively via making all the entries zero except the rows corresponding to $\mathcal{S}$. Consequently, the learned dictionary $\Phi_{\mathbf{B}}(\mathbf{U})$ can partially reconstruct the unseen time-series $\mathbf{Z}$ for the subset $\mathcal{S}$ of its dimensions, if $\mathcal{J}^{\mathcal{S}}_{rec}(\mathbf{Z}, \mathbf{B}, \mathbf{U})$ is relatively small. Provided a small reconstruction error $\mathcal{J}^{i}_{rec}(\mathbf{Z}, \mathbf{B}, \mathbf{U})$ is present, we can look at component-wise class assignments: the i-th dimension of $\mathbf{Z}$ can be assigned to the class q from $\mathbf{X}$ which has the highest contribution to the reconstruction of this dimension of $\mathbf{Z}$, as $q = \arg\max_q \sum_{j:\boldsymbol{l}_j=q} \sum_{r=1}^{k} \beta_{ir} u_{jr} \gamma_r$.

3.3 Incremental Clustering of Unseen Time-series

Based on the discussion of section 3.2 and relying on the partial similarity of different MTS classes and descriptive quality of the learned attributes of the dictionary, we propose Algorithm 1. This algorithm incrementally clusters the unseen sequences from $\mathcal{Z}$ into a dendrogram $\mathcal{H}$ as they are observed in an online fashion, as well as finding potential sub-clusters among them. To that aim, for each unknown MTS sequence $\mathbf{Z}$ we prepare an encoding matrix $\mathbf{R} \in \mathcal{R}^{N \times f}$, i-th column of which represents the weights of contribution from $\mathbf{X}$ in the reconstruction of the i-th dimension of $\mathbf{Z}$. Therefore,

[1] https://github.com/bab-git/MKD_Unseen_MTS

Algorithm 1: Incremental Clustering of an Encoded MTS data

Input: $\mathbf{R}$: Calculated encoding matrix of the unseen data $\mathbf{Z}$, $\mathcal{H}$: The hierarchical tree.
Output: Place of $\mathbf{Z}$ in the hierarchy $\mathcal{H}$.

If $\exists C_n$ such that $d(\mathbf{Z}, C_n) \leq \bar{d}(C_n)$ **then**
  If C_n is a leaf node **then** add $\mathbf{Z}$ to C_n;
    If $(\bar{d}(C_{n1}) + \bar{d}(C_{n2}))/2\bar{d}(C_n) \leq k_{clust}$ **then**
      split C_n into C_{n1} and C_{n2} using k-means;
      If $(\bar{d}(C_{n1}) + \bar{d}(C_{n2}))/2\bar{d}(C_n) \leq k_{rmv}$ **then**
        Replace C_n with C_{n1} and C_{n2};
        Re-cluster $\{C_{n1}, C_{n2}\}$ with same branch leaves;
      else add $\{C_{n1}, C_{n2}\}$ as the children of C_n;
  else Create a new child for C_n as C_{n_t} and add z to it;
else Create a new leaf at the top level containing $\mathbf{Z}$;

Table 1: Information regarding the selected datasets for the experiments.

Dataset	# sequences	# classes	# dimensions	unseen classes
Cricket Umpire	180	12	6	*last-hour*, *six*, *no-ball*, *Out*
CMU mocap	200	20	62	*circle*, *Punch*, *throw*, *Walk*, *lift*
Articulatory Words	575	25	9	5-fold cross-validation splits
Squat	81	3	142	*Prepare*, *Go down*

$r_{ji} = \sum_{t=1}^{k} \beta_{it} u_{jt} \gamma_t$ where r_{ji} denotes the j-th entry of the i-th column of $\mathbf{R}$. This matrix is considered as a rich encoded descriptor for dimensions of $\mathbf{Z}$ based on $\mathbf{X}$ and is used in Algorithm 1 to compare $\mathbf{Z}$ to the previously categorized unseen data in $\mathcal{H}$ to find the best place for $\mathbf{Z}$ in the dendrogram. Line 1 of the algorithm finds C_n as the most similar node to $\mathbf{Z}$ based on the distance term $d(\mathbf{Z}, C_n) = \|\mathbf{R}_{\mathbf{Z}} - \overline{\mathbf{R}}_{C_n}\|_F^2$, and the intra-cluster distance for each node C_n as $\bar{d}(C_n) = E_{\mathbf{Z}_i \in C_n}[d(\mathbf{R}_{\mathbf{Z}_i}, \overline{\mathbf{R}}_{C_n})]$, where $\overline{\mathbf{R}}_{C_n} = E_{\mathbf{Z}_i \in C_n}[\mathbf{R}_{\mathbf{Z}_i}]$.

In line 7, each of $\{C_{n1}, C_{n2}\}$ is merged with the best C_m from the same branch to form C_{m+ni} which satisfies $2\bar{d}(C_{ni+m})/[\bar{d}(C_{ni}) + \bar{d}(C_m)] \leq k_{clust}$ for $i = 1, 2$.

Although, one might want to tune the parameters $\{k_{rmv}, k_{clust}\}$ (e.g. via doing a cross-validation on $\mathbf{X}$), in practice, choosing $k_{rmv} = 0.3$, $k_{clust} = 0.5$ gives an acceptable clustering result.

4 Experiments

To evaluate the performance of our sparse coding framework for representation and discrimination of unseen data, we choose the MTS datasets Cricket Umpire, CMU mocap, Articulatory Words, and Squat with the descriptions provided by [7] as well as in Table 1. For all the datasets, to compute the kernel matrices for individual dimensions of the data, we use the Gaussian kernel $\mathcal{K}_l(\mathbf{X}_i, \mathbf{X}_j) = exp(-\mathcal{D}_l(\mathbf{X}_i, \mathbf{X}_j)/\delta_l)\ \forall l = 1 \dots f$, where $\mathcal{D}_l(\mathbf{X}_i, \mathbf{X}_j)$ is the computed pairwise DTW-distance [1] (can be substituted with any other data distance) between the l-th dimension of $\mathbf{X}_i$ and $\mathbf{X}_j$. Parameter δ_l is set equal to the average of $\mathcal{D}_l(\boldsymbol{x}_i, \boldsymbol{x}_j)$ distances over all the data samples. To make sure that $\mathcal{K}_l(\mathcal{X}, \mathcal{X})$ remains PSD, we use the common clipping operation by setting its negative eigenvalues to zero. For the tuning of (λ, T_0) in Eq. 2, we use 5-fold cross-validation based on its objective value. Also, the dictionary size is determined as $k = \{\#\text{seen classes}\} \times T_0$.

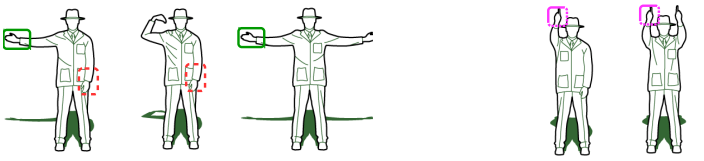
4.1 Partial Reconstruction of Multivariate Time-series

In order to evaluate the reconstruction quality for each unseen data $\mathbf{Z}$, we define the dimension-reconstruction accuracy measure as $DRA := \frac{\#\text{ dimensions that } \{\mathcal{J}_{rec}^i(\mathbf{Z},\mathbf{B},\mathbf{U}) \leq 0.1\}}{\#\text{ total dimensions}}$ using Eq. 4. In addition, each reconstructed dimension of $\mathbf{Z}$ which satisfies the above threshold is interpreted via the class of data with the most contribution as in Sec. 3.2.

We select Cricket dataset to demonstrate how the proposed MKD-SC framework reconstructs the unseen classes in the dimension-level, and Table 2 and Figure 2 report the reconstruction quality and the interpretation of the unseen classes respectively. Based

Table 2: DRA measure (%) for unseen Cricket classes, and their partial descriptions based on the training classes based on the right **(R)** or left **(L)** hands.

Unseen Classes	no-ball	last-hour	Out
DRA (%)	100	50	50
Contributed Classes	penalty (L), short (L), wide (R)	wide (L)	six (R)



(a) $no\ ball\ \rightarrow \{short+\ \ wide\}$ (b) $out \rightarrow \{six\}$

Fig. 2: Dimension-level interpretation of *no-ball* and *out* (Cricket) based on the training classes. Related dimensions are specified via using same-color rectangles.

on the results, the *No ball* class is fully reconstructed via its partial relation to the movement of the left hand in the *Short* class and to the right hand in the *Wide* class.

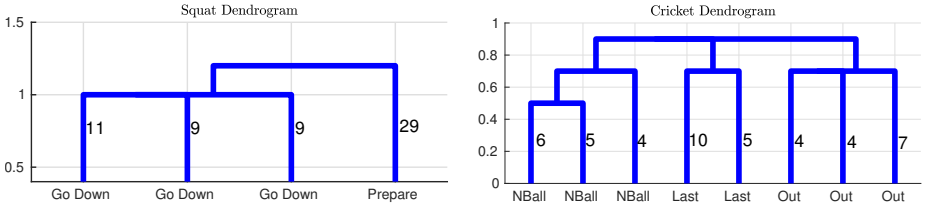
4.2 Incremental Clustering of the Unseen Time-series

The proposed incremental clustering is implemented on the unseen categories of Table 1 after computing the $\mathbf{R}$ matrix based on the sparse code of each unseen data (Sec. 3.3). We use the average clustering error (CE) [10], and normalized mutual information (NMI) [3] for quantitative evaluation of the clustering results. To that aim, we cut each dendrogram from where it has the equal number of clusters to the ground truth.

Baselines: To evaluate the quality of attributes which are learned by MKD-SC framework, as the most relevant method we choose self-learning algorithm [11] without its novelty detection part. In addition, we apply the spectral clustering algorithm on the original kernel matrix $\mathcal{K}(\mathbf{Z}, \mathcal{X})$ to compare our framework to the normal clustering of $\mathcal{Z}$. As another baseline, we also use the NNKSC algorithm [7] as the single-kernel predecessor of MKD-SC, for which the $\mathbf{R}$ matrix becomes an N-dimensional vector.

Based on clustering dendrograms in Figure 3, the unseen Squat and Cricket classes are well categorized, which shows the effectiveness of the learned attributes for distinct representation of the unknown classes. The incremental clustering also categorized these unseen classes into a few sub-clusters. For Squat (Fig. 3-a), the 3 sub-clusters for the *Go down* class are related to different performance styles of the dataset's 3 participants regarding this specific phase of the squat. Similarly, for each unseen category of the Cricket dataset (Fig. 3-b), there are sub-clusters recognized for each of the distinct main clusters which shows the existing structured variation within each of this classes.

According to the clustering results in Table 3, the proposed MKD-SC method provides encodings which lead to better clustering of the unseen data compared to the baselines. The superiority of the original kernels over NNKSC and self-learning methods depends on the discriminative quality of the original kernels, for instance for the Cricket dataset. Self-learning method can have a better performance than NNKSC and Kernel-based baselines if its descriptor-based features can better discriminate between the different categories of the unseen classes.



(a) (b)

Fig. 3: Incremental clustering result for unseen classes of Squat **(a)** and Cricket **(b)**.

Table 3: Clustering error (CE) (%) and NMI the unseen categories.

Methods	Words		Squat		CMU		Cricket	
	CE	NMI	CE	NMI	CE	NMI	CE	NMI
MKD-SC(Proposed)	12.31	0.89	0	1	9.28	0.92	0	1
Self-learning [11]	18.75	0.84	0	1	14.25	0.87	16.63	0.85
NNKSC[7]	21.61	0.78	15.74	0.88	18.88	0.85	12.45	0.87
Spectral Clustering	27.51	0.76	13.04	0.90	23.45	0.76	8.04	0.89

5 Conclusion

In this research, we proposed an unsupervised framework which provides interpretable analysis of unseen classes in MTS datasets. We proposed a novel multiple-kernel dictionary structure based on individual mappings of the MTS dimensions into the feature space represented by dimension-based kernel functions. The MKD uses weighted combinations of these kernels along with selections from training data in order to learn its compact descriptive attributes (bases). Based on the learned attributes, our unsupervised MKD-SC framework reconstructs the unseen classes (partially/fully) in the feature space according to overlapping of their dimensions with those of the seen categories and provides an interpretable description of the novel data. Based on the obtained sparse encodings, we proposed an incremental clustering to gradually categorize novel MTS, and to form a dendrogram. Experiments on MTS benchmarks showed that our MKD-SC framework can describe the dimensions of the unseen classes via their relation to those of the seen ones. In addition, the incremental clustering provides better clustering accuracy comparing to the baselines and also reveals the existing sub-clusters in the data.

References

1. Adistambha, K., Ritz, C., Burnett, I.: Motion classification using dynamic time warping. In: MMSP'08 Workshop. pp. 622–627 (Oct 2008)
2. Alabdulmohsin, I., Cisse, M., Zhang, X.: Is attribute-based zero-shot learning an ill-posed strategy? In: ECML/PKDD'16. pp. 749–760. Springer (2016)
3. Ana, L., Jain, A.K.: Robust data clustering. In: IEEE Conference on Computer Vision and Pattern Recognition. vol. 2, pp. II–128. IEEE (2003)
4. Cheng, H.T., Sun, F.T., Griss, M., Davis, P., Li, J.: Nuactiv: Recognizing unseen new activities using semantic attribute-based learning. In: MobiSys'13. pp. 361–374. ACM (2013)
5. Hosseini, B., Hammer, B.: Localized multiple-kernel sparse coding for discriminative representation and feature selection. In: Submitted to ICDM'18 (2018)
6. Hosseini, B., Hammer, B.: Non-negative quadratic pursuit. vixra pre-print (2018)
7. Hosseini, B., Hülsmann, F., Botsch, M., Hammer, B.: Non-negative kernel sparse coding for the analysis of motion data. In: ICANN'16. pp. 506–514. Springer (2016)
8. Jian, L., Xia, Z., Liang, X., Gao, C.: Design of a multiple kernel learning algorithm for ls-svm by convex programming. Neural Networks **24**(5), 476–483 (2011)
9. Lampert, C.H., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. In: CVPR'09. pp. 951–958. IEEE (2009)
10. Lu, C., Feng, J., Lin, Z., Mei, T., Yan, S.: Subspace clustering by block diagonal representation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2018)
11. Lu, D., Guo, J., Zhou, X.: Self-learning based motion recognition using sensors embedded in a smartphone for mobile healthcare. In: WASA'16. pp. 343–355. Springer (2016)
12. Peng, P., Tian, Y., Xiang, T., Wang, Y., Pontil, M., Huang, T.: Joint semantic and latent attribute modelling for cross-class transfer learning. TPAMI **40**(7), 1625–1638 (2018)
13. Qiu, Q., Jiang, Z., Chellappa, R.: Sparse dictionary-based representation and recognition of action attributes. In: ICCV'11. pp. 707–714. IEEE (2011)
14. Rubinstein, R., Zibulevsky, M., Elad, M.: Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit. Cs Technion **40**(8), 1–15 (2008)
15. Socher, R., Ganjoo, M., Manning, C.D., Ng, A.: Zero-shot learning through cross-modal transfer. In: Advances in neural information processing systems. pp. 935–943 (2013)
16. Zhang, Z., Saligrama, V.: Zero-shot learning via semantic similarity embedding. In: Proceedings of the IEEE international conference on computer vision. pp. 4166–4174 (2015)

On the Feasibility of Deep Belief Networks for Tool Wear Monitoring in CNC Machines

Aakanksha Bapna, Naveen Kumar Thokala, Kriti Kumar, and M. Girish Chandra

TCS Research & Innovation, Bengaluru, Karnataka, India-560066
{aakanksha.bapna,naveen.thokala,kriti.kumar,m.gchandra}@tcs.com

Abstract. The quality of the components is of utmost importance especially in high precision engineering and aerospace industries. The machining tools wear with time and can result in poor quality components if not monitored carefully. The manufacturing industry is moving towards automation i.e Industry 4.0, which mandates most of the tasks to be automated. So, there is a need for data-driven models to match the best operator's performance for (i) continuous monitoring of tool wear so that component quality is not compromised and (ii) rendering tool change alerts at the correct time. In this paper, it is aimed to use Deep Belief Networks(DBNs) to model tool-wear using the data from internal sensors like spindle load and axis positions. The amount of data specific to a tool that is available from the Computer Numerical Control (CNC) machines under study is very limited. This inhibits the use of deep architectures for tool wear estimation as it requires a huge amount of data to train such networks. Additional data is therefore generated using Restricted Boltzmann Machines (RBM) based generative models. Thereafter, a DBN with regression layer is used to extract robust features from the combined data and to predict Remaining Useful Life (RUL) of the tool. The performance of the proposed approach for estimating the tool wear is on par with that achieved using the handcrafted features.

Keywords: Generative Models · Tool Wear Monitoring · Regression

1 Introduction

The fourth Industrial revolution has invoked a surge in automation of numerous tasks to make factories smart. The application defined in this paper is towards the automation of CNC machines operation in a high precision engineering industry. The sensory information like spindle positions, motor currents, temperature of the coolant etc., can be judiciously processed for automating a number of tasks like tool wear detection, overcurrent fault detection, assessment of the quality of the component, machine malfunctioning and many more useful tasks. Among these, tool wear estimation is an important aspect of the manufacturing process to maintain the quality of the product. Tool wear, if not detected in time either due to operator negligence or inaccurate decision making, can damage both the component as well as the processing machine. Sensor data analytics would help

in accurate tool wear detection and maintaining the quality of components with less operating cost, even when there are changes in a machine, material, cutting condition over time. Two problems that make the tool wear estimation challenging in the considered scenario are: (i) Unable to install external sensors like accelerometers: according to the state of art, vibration data is most favourable for identifying the condition of the tool [1]. Since it was not permitted to put any external sensors in the CNC machines we studied, the analytics was carried out using already available internal sensors only. (ii) Lack of huge amount of data specific to a tool: there is a significant variation in the type of components manufactured and operations executed to manufacture each component. Since a different tool is used for each kind of operation, the data for a specific type of tool is very scarce.

In order to monitor the condition of a tool, we need to model Remaining Useful Life (RUL) of the tool as a function of input parameters contained in the limited data available for every operation. In the traditional machine learning approach, an expert would derive the features based on systems understanding and also by carrying out a preliminary examination of data. In this process, there is a possibility of information loss coupled with the difficulty in arriving at features robust to minor variations in the data. Apart from this, extraction of features from every new kind of operation is quite difficult and non-trivial. Deep architectures can address these kinds of issues [2].

As mentioned, the major challenge in applying deep architectures for the problem under consideration is that a large amount of data required for training these networks is not available. Therefore, Restricted Boltzmann machines (RBMs) are used for modelling the data distribution and in turn used to generate additional data to train deep networks. The generated data is combined with existing data to facilitate the use of deep networks.

In 2006, Hinton [3] discovered that much better results could be achieved in

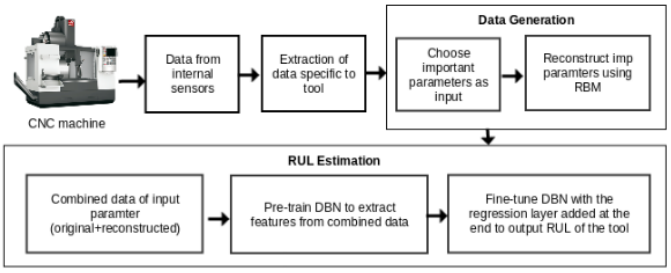


Fig. 1. Block Diagram of the approach taken for estimating RUL of the tool.

deeper architectures when each layer (RBM) is pre-trained with an unsupervised learning algorithm (Contrastive Divergence). Hence, Deep Belief Networks (DBNs) are explored in this work to extract useful features from the given data and then, a regression layer is added over the last layer of DBN as shown in Fig-

ure 3. The entire network is trained in a supervised way using backpropagation errors calculated using the actual regression outputs in order to fine-tune the weights.

Related Work: There are many works [4] [5] which review the different approaches for RUL estimation. Cubillo et al. [4] reviews the concept of physics-based models for prognostics. Tsui et al. and Si et al. [6][7] proposed data-driven algorithms for RUL estimation. Zhang et al. [8] and Guo et al. [9] proposed deep learning based techniques for RUL estimation. As explained, our approach considerably differs from the earlier works in dealing with the lack of data problem when using deep networks.

2 Methodology used for RUL Estimation

The CNC machines we studied manufacture many different kinds of precision engineering components. To manufacture a component, a workpiece undergoes multiple operations like roughing, drilling, milling etc. For every kind of operation a different type of tool was used. Hence to analyse a particular type of tool used for an operation it was necessary to filter out the data for every part number [1] of a component undergoing that operation and then identify the changes in the pattern of the data with every part manufactured. Each and every operation is unique and affects the loading on different axis motors (power consumed by the axis-wise servomotors) based on the job it performs; for example, $SpindleLoad$ and $Feed_z$ varies a lot for pocket rough operation and other axis servo loads (x & y-axis) are almost constant. For other operations like boss finish and boss rough, $SpindleLoad$ is almost constant and other axis servo loads have some variations which would be useful in detecting tool wear. So, correlation analysis was used to identify the appropriate features to be used for modelling the tool wear for a specific operation.

In order to carry out tool condition monitoring, the sensor data is segregated for every tool by filtering the operation and component type. In this paper, we considered the pocket rough operation in great detail to validate the proposed methodology. As per the data sheets of the tool used for pocket rough operation, operation time and tool life are 1450 seconds and 15 parts respectively. However, the operator rather continues to use the same tool even after its mentioned tool life until he thinks the tool is not in a good condition based on his expertise. This is done to reduce the operational cost of the manufacturing plant. At the same time, loosing on the component's quality results in it's scraping which is a bigger loss, so the operator takes an 'optimal' decision to change the tool based on his expertise, maximising both the tool life and quality of the component.

To capture the operator's actions for predicting the tool condition, RUL modelling was carried out on the different tools used for various operations. As mentioned, the inputs to the RUL modelling algorithm were selected based on the correlation analysis; for pocket rough operation the inputs are raw data from the

[1] **Note:** Part wherever user in this paper refers to the unique make of a component. A part number is the serial number of that part.

sensors measuring the $SpindleLoad$ and $Feed_z$ and output of the network is the Remaining Useful Life (RUL) of the tool. Tool change records (time instants) were used to create a Remaining Useful Life (RUL) vector. We undertook the following methodology to learn a regression model to estimate the RUL of the tool as a function of $SpindleLoad$ and $Feed_z$ using the Deep Belief Networks (DBNs):

2.1 Generating Data

As mentioned in the Section 1, the actual data from the manufacturing plant is not sufficient enough to train the deep architectures. Therefore, RBMs were used to generate more data from the existing sensor data ($SpindleLoad$,$AxiswiseFeed$, $AxisServoLoads$ etc.). But, rather than using all the sensor data for data generation, correlation analysis was used judiciously to select only those input parameters having good correlation with the RUL of the tool and used for data generation. This helps in improved performance of data generation process as modelling the data distribution with too many parameters and less data is not very accurate. For pocket rough operation, $SpindleLoad$ and $Feed_z$ were selected to use in data generation and then the actual data along with the generated data were used to learn the DBNs.

Restricted Boltzmann Machines are undirected probabilistic graphical mod-

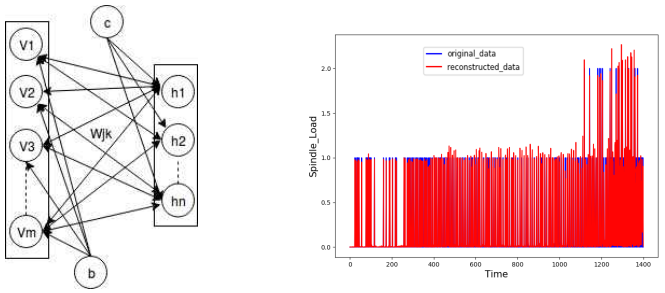


Fig. 2. (left) Network depicting the sturcture of an RBM, (right) data reconstruction using RBM (original data is represented in blue and reconstructed data in red).

els which can be interpreted as a class of stochastic neural networks. RBMs are used to characterize the probability distribution of the data. For a given data, representations of the data in different dimensional space can be obtained which could provide us with additional information about the data. Then, the hidden representations could be used to generate the data patterns similar to input data. RBMs consists of two layers, visible layer and the hidden layer as shown in Figure 2. The goal is to maximize the joint probability distribution of visible and hidden layer i.e $P(V, H)$ and is defined using energy based model as shown in Equation 1:

$$P(v, h) = 1/Z * exp\{-E(v, h)\}, \tag{1}$$

where Z is the partition function and $E(V, H)$ is the energy of the network which is defined as

$$E(v, h) = -(b^T v) - (c^T h) - (v^T W h), \tag{2}$$

where W are the weights connecting the visible layer and the hidden layer, b is the bias for the visible layer and c is the bias for hidden layer. To maximize the probability distribution, the energy of the network has to be minimised.
As well known, the direct evaluation of $P(v, h)$ is difficult, but conditional probabilities $P(h/v)$ and $P(v/h)$ are easy to compute and sample from. The joint probability distribution $P(h, v)$ can be estimated from the conditional distributions. The conditional distribution of a hidden neuron h_j in a RBM is represented as a sigmoid of the weighted sum of all visible neurons $(v_1..v_k)$ and the conditional distribution of visible units is also obtained similarly. These probabilities as described below:

$$P(h_j = 1|v = x; w) = \sigma\Big(\sum_{k=1}^{m} w_{kj} v_k + c_k\Big) \tag{3}$$

$$P(v_k = 1|h; w) = \sigma\Big(\sum_{j=1}^{n} w_{jk} h_j + b_j\Big) \tag{4}$$

The weights (w) and biases (b and c) of the network are learnt using the Contrastive Divergence algorithm [3]. For the problem at hand, the visible units were fed with the input data patterns $v = x$ ($SpindleLoad$ and $Feed_z$) to learn the hidden representation. In order to use binomial RBMs for real valued sensor data like in our case, the data was normalised in the range [0,1]. The learned hidden representations of the RBM network were used to reconstruct data samples (x') from different input data patterns x. Figure 2 shows the reconstruction of the spindle load using RBMs.
Details of the RBM network used for input data generation are: number of neurons in the hidden layer was chosen to be half of the input size as that worked best for data generation. Upon experimentation, it was found that a high learning rate (0.1) and at least 100 training epochs are necessary for getting good reconstructions as shown in Figure 2. The combined data used for training deep networks consisted of 50% original data and 50% generated data.

2.2 Deep Belief Networks

A DBN consisting of three RBM hidden layers and an output regression layer as shown in the Figure 3 is used to estimate the RUL of a tool from the sensor data ($SpindleLoad$ and $Feed_z$) of an operation . Performing regression using DBNs consists of 2 stages:

1. **Pre-training**: It consists of learning the weights of the stacked RBM layers from the input data using Contrastive Divergence which is carried out in an unsupervised manner [3]. Each layer is learnt one by one. This step is useful in extracting robust features from the input sensor data.

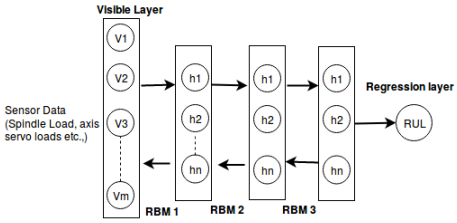


Fig. 3. A DBN with 3 hidden layers (connections between which are bi-directional) and last layer is the regression layer.

2. **Fine-tuning**: The learning of entire network (DBN + regression layer) takes place in this step. The weights of the stacked RBM layers are initialised with the ones learnt in the last step. Training of the network is carried out in a supervised fashion using both input and output data through gradient descent based backpropagation of errors.

We experimented with different network architectures and made some useful observations. In the pre-training stage, multiple layers of representations were learnt and care was taken not to over-fit in the process of achieving very good reconstruction of the input data. In the Fine-tuning stage, these weights were tuned appropriately after adding the output regression layer with single neuron outputting RUL.
Details of DBN used: consists of 3 hidden layers with 1400,700 and 1400 neurons in each layer respectively. The remaining hyper-parameters which worked well for estimating RUL for the given data are mini-batch size=10, learning rate for pre-training=0.01, number of epochs for pre-training=20, learning rate for fine-tuning=0.01, number of epochs for fine-tuning=200.

3 Results and Discussion

In order to test the feasibility of DBNs for RUL estimation in CNC machines, we compared the performance of DBN with the Support Vector Regression(SVR) model using features derived by the traditional analysis. Features like frequency of distinct amplitudes in the SpindleLoad waveform were used to model the RUL. Support Vector Regression(SVR) with radial basis kernel was used to estimate the RUL as this kernel was found useful in modelling the complex relationships. Three methods were explored for RUL modelling - SVR with the original data, SVR with combined (real+generated) data and DBNs with combined data. All these algorithms were tested on two different kinds of tools used in pocket rough operation for manufacturing different components. The results for both the types of the tools- Type-A and Type-B used for pocket roughing operation on machine A and machine B are captured. The data for tool Type-A consisted of seven instances of tool change and k-fold validation was used to test the efficacy of the algorithms. In order to achieve this, the data for a tool (one of the 7 tools of say, Type-A) is kept aside for testing in every iteration. The generated data is also

partitioned similarly. Mean Square Error(MSE) and Mean Absolute Percentage Error (MAPE) are used as error metrics for evaluating the performance of these regressors. Table 1 and Table 3 represent the average of MSE and MAPE obtained upon testing on 7 tools each for Type-A and Type-B. It is also important

Table 1. RUL prediction errors for Type-A Tool.

Method	MSE	MAPE
Engineered-features+SVR	5.11	41.26
Engineered-features+SVR (with generated data)	5.87	38.67
Deep Belief Network (with generated data)	4.79	23.42

Table 2. RUL Stage-wise Accuracy for Type-A Tool.

Method	RUL15	RUL10	RUL5
Engineered-features+SVR	91	79	78
Engineered-features+SVR (with generated data)	90	83	80
Deep Belief Network (with generated data)	74	85	77

Table 3. RUL prediction errors for Type-B Tool .

Method	MSE	MAPE
Engineered-features+SVR	4.51	33.74
Engineered-features+SVR (with generated data)	3.93	28.46
Deep Belief Network (with generated data)	4.53	25.05

Table 4. RUL Stage-wise Accuracy for Type-B Tool.

Method	RUL15	RUL10	RUL5
Engineered-features+SVR	86	88	87
Engineered-features+SVR (with generated data)	85	91	90
Deep Belief Network (with generated data)	87	89	77

to analyse the accuracy of the predictions at different stages of tool life in order to know which algorithms perform better in early and finishing stages of tool life. This analysis address different requirements like inventory management, alerts to the operator for tool change etc. The error metrics evaluated at different stages of tool-life (RUL-15,10 and 5) are shown in Table 2 and Table 4. It is evident from figures and tables that the performance of DBN is on par or better than the SVR algorithm modelled using derived or handcrafted features. It is worth noting that compared to [8], where an advanced version of DBNs is used for RUL estimation for the C-MAPPS dataset which is a simulated dataset, we have considered the real data. Also, useful to note is the fact that the performance of the SVR based algorithm is also improved by incorporating the generated data.

4 Conclusion

In this paper, the RUL estimation using Deep Belief Networks (DBNs) on real and limited amount of data has been systematically addressed. The proposed methodology performed better than the conventional machine learning (ML) algorithms involving engineered features. Apart from improved accuracy in prediction, DBNs can lessen the burden of deriving features by the domain-expert for every new operation or tool used. The improved accuracy in estimating RUL of the machining tool also helps in the efficient utilization of the machining tools while maintaining the quality of the manufactured component.

References

1. Javed, Kamran, et al. "Enabling health monitoring approach based on vibration data for accurate prognostics." IEEE Transactions on Industrial Electronics 62.1 (2015): 647-656.
2. Mhaskar, Hrushikesh, Qianli Liao, and Tomaso A. Poggio. "When and why are deep networks better than shallow ones?." AAAI. 2017.
3. Hinton, Geoffrey E., Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. Neural computation 18.7 (2006): 1527-1554.
4. Cubillo, Adrian, Suresh Perinpanayagam, and Manuel Esperon-Miguez. "A review of physics-based models in prognostics: Application to gears and bearings of rotating machinery." Advances in Mechanical Engineering 8.8 (2016): 1687814016664660.
5. Jardine, Andrew KS, Daming Lin, and Dragan Banjevic. "A review on machinery diagnostics and prognostics implementing condition-based maintenance." Mechanical systems and signal processing 20.7 (2006): 1483-1510.
6. Tsui, Kwok L., et al. "Prognostics and health management: A review on data driven approaches." Mathematical Problems in Engineering 2015 (2015).
7. Si, Xiao-Sheng, et al. "Remaining useful life estimation a review on the statistical data driven approaches." European journal of operational research 213.1 (2011): 1-14.
8. Zhang, Chong, et al. "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics." IEEE transactions on neural networks and learning systems 28.10 (2017): 2306-2318.
9. Guo, Liang, et al. "A recurrent neural network based health indicator for remaining useful life prediction of bearings." Neurocomputing 240 (2017): 98-109.

c-RISE: contract random interval spectral ensemble for time series classification

Michael Flynn, Jason Lines, and Anthony Bagnall

University of East Anglia, Norwich Research Park, UK ajb@uea.ac.uk
http://www.timeseriesclassification.com

Abstract. The Random Interval Spectral Ensemble (RISE) is a recently introduced tree based ensemble time series classification algorithm where each tree is built on a distinct set of Fourier, autocorrelation and partial autocorrelation features. It is a component in meta ensemble HIVE-COTE [8]. RISE has run time complexity of $O(nm^2)$, where m is the series length and n the number of train cases. This is prohibitively slow when considering long series, which are common in problems such as audio classification where spectral approaches are likely to perform better than classifiers built in the time domain. We propose an enhancement of RISE that allows the user to specify how long the algorithm can have to run. The contract RISE (c-RISE) allows for check-pointing and adaptively estimates the time taken to build each tree in the ensemble through learning the constant terms in the run time complexity function. We show how we can accurately estimate run time in this way, and demonstrate the flexibility this approach offers a new case study involving classifying bird calls.

1 Introduction

The most successful approaches to time series classification (TSC), the classification of real valued, ordered series, are based on transformation and ensembling. The most accurate algorithm, HIVE-COTE [8], is an ensemble of five separate representation/classifier combinations. One of the component representations is based in the frequency domain and uses an interval based ensemble called the Random Interval Spectral Ensemble (RISE). The premise behind RISE is that the discriminatory features are in the frequency domain, but that the relevant area of the series may be embedded in irrelevant and confounding noise. Suppose, for example, we wish to detect whether a certain type of bird call is present in an audio clip. We do not know whether the call is in all of the clip or whether it appears at any time in a small segment of the overall series. We could attempt to preprocess the audio to detect the region of interest, but this requires domain knowledge, an effective model of the type of audio to be detected and the setting of numerous parameters. Our interest is in a fully automated approach. We assume we have series measured over arbitrary time, i.e. that the segmentation is unsupervised. This means the series may be of longer duration than the signal of interest. e.g. the bird calls we want to classify may occur in only a part of the

signal. RISE compensates for this characteristic by taking random intervals and building a classifier on each interval. For any given interval, rise takes the Fourier transform and finds the autocorrelation and partial autocorellation function (ACF and PACF), then constructs a decision tree on these features.

RISE was shown to be significantly better than other spectral based approaches on the TSC archive data and on simulated data [2]. However, RISE is computationally expensive, since each transformed series is based on an $O(m^2)$ operation (finding the PACF), where m is the series length. Audio signals in particular tend to be very long. Our aim is to make RISE more useful by improving the runtime and making it a contract classifier, i.e. a classifier where you can specify approximately the amount of computational time allowed to build the model. In Section 2 we describe existing approaches to spectral based classification, including RISE. In Section 3 we highlight the changes we have made to RISE in order to form the contract version, c-RISE. We then assess the impact of restricting time for RISE on classification accuracy in Section 4, before concluding in Section 5.

2 Background

Time series classification spans a very wide range of problems, including image outlines, ECG, sensor data, human activity recognition and spectrograph. The majority of TSC algorithms operate in the time domain, finding either phase dependent or phase independent features. However, in many areas, such as sound classification, discriminatory features are found in the frequency domain rather than the time domain. Time based classifiers will perform poorly on this type of problem. Common approaches for solving problems such as this involve transforming the data into power spectrum or using autocorrelation-based features (for example, [6, 1]). However, one problem with this is that performing a Fourier transform on the whole series may be inappropriate. Particularly for very long series, the discriminatory features may be embedded within confounding noise. One approach is to use a sliding window to make the problem into a multivariate time series classification problem. However, this approach has certain difficulties, such as setting window size, and there is still the problem of finding relevant features in the massively expanded signal. The Random Interval Spectral Ensemble (RISE) takes a univariate approach.

2.1 The Random Interval Spectral Ensemble (RISE)

RISE draws on ideas from tree-based ensembles such as random forest [5] and the TSC interval feature classifier time series forest (TSF) [7]. Like TSF, we build trees on random intervals from the data to construct a random forest classifier. A key difference is that TSF uses time domain features by calculating the mean, variance, and slope of each interval, but RISE extracts spectral features over each random interval instead. Once we have derived the spectral features, we build a decision tree using the random tree algorithm used by random forest.

The base RISE algorithm is described in Algorithm 1. The first tree in RISE is a special case that uses the whole series. We include this step for continuity with the previous spectral classifiers used in COTE which only used the whole series. However, for very long series it may be impractical to transform the whole series and we will review this element in future work. The procedure for building RISE is outlined in Algorithm 1.

Algorithm 1 BuildRISE(Training data *train*, number of classifiers r, minimum interval length *minLen*)

Let $\mathbf{F} \leftarrow < F_1 \dots F_r >$ be the trees in the forest.
Let m be the length of series in *train*
$wholeSeriesFeatures \leftarrow$ getSpectralFeatures(*train*)
buildRandomTreeClassifier($\mathbf{F}_1$,*wholeSeriesFeatures*)
for $i \leftarrow 2$ to r **do**
    $startPos \leftarrow$ randBetween$(1, m - minLen)$
    $endPos \leftarrow$ randBetween$(startPos + minLen, m)$
    $interval \leftarrow$ removeAttributesOutsideOfRange(*train*,*startPos*,*endPos*)
    $intervalFeatures \leftarrow$ getSpectralFeatures(*interval*)
    buildRandomTreeClassifier($\mathbf{F}_i$,*intervalFeatures*)

RISE uses several forms of spectral features: the power spectrum, the autocorrelation function, the partial autocorrelation and the autoregressive model. New classes are classified using a simple majority vote. Further details can be found in [8].

3 Contract RISE (c-RISE)

In many areas it may be advantageous or even necessary to constrict the run time of a classification algorithm. Generally, it not well understood how long classification algorithms take to run for a given problem. It is of practical importance when considering which algorithm to use or how much preprocessing to perform. This is a particular difficulty when using cloud services in which the computation is charged for per time period, or situations in which there is a hard deadline or where there is a limit on how long a process is allowed to run. Two solutions to these problems are check-pointing, periodically saving a partial version of the classification model to disk, and contracting, limiting the the amount of computational time an algorithm is allowed. Used together, they make a classifier more flexible and useful to the practitioner. Check-pointing RISE is easy, especially with a Java implementation; we can simply serialise the constructed trees at certain points and adapt RISE to allow the loading from file. There is no point check-pointing small problems, so we configure RISE to check-point approximately once every 24 hours. Hence both check-pointing and contracting RISE require the ability to control the runtime of building a single tree.

RISE performs three transformations: A discrete Fourier transform (DFT) to find the power spectrum, construction of the autocorrelation function (ACF), and derivation of the partial autocorellation function (PACF) from the ACF. With the simplest implementation, each of these is $O(r^2)$, where r is the interval width. We can improve the efficiency of the Fourier transform to $O(rlog(r))$ by using the fast Fourier Transform (FFT). To gain the full benefit we restrict RISE to intervals of length the power of 2. However, the best average case complexity for the ACF and PACF are $O(r^2)$. Hence, the transformations will dominate the runtime in relation to the decision tree, so we can reasonably model the runtime t for a single member of the ensemble of interval length r as

$$t = a \cdot r^2 + b \cdot r + c.$$

If we fix the contract time t and knew the constant factors a, b and c we could find the positive root of the quadratic and use that as the maximum allowable interval for the tree. The quadratic terms will of course be both problem and hardware dependent. Hence, we use an adaptive algorithm to learn these parameters. For each tree we build, we record the selected interval and the observed run time. Using these data, we refit a least squares linear regression model. For clarity, we let $x_1 = r_1^2$ and $x_2 = r_1$, our dependent variable matrix is then

$$X = \begin{bmatrix} 1, x_{11}, x_{12} \\ 1, x_{21}, x_{22} \\ \dots \\ 1, x_{k1}, x_{k2} \end{bmatrix}$$

the estimates of the parameters are $B = (\hat{a}, \hat{b}, \hat{c})^T$ and our response variable is $Y = (y_1, y_2, \dots, y_k)^T$. The least squares estimates are then

$$B = (X^T X)^{-1} X^T Y.$$

Since $(X^T X)$ is based on sums of squares, we do not have to recalculate if from scratch each time. It is also possible to update $(X^T X)^{-1}$ online with the Sherman-Morrison formula, but we leave that for future work. After the construction of each tree, we update the remaining contracted time t, re-estimate the coefficients $t = \hat{a} \cdot r^2 + \hat{b} \cdot r + \hat{c}$, and calculate a new maximum allowable interval r. This is used as the maximum for the next iteration. The adapted RISE, c-RISE, is described in Algorithm 2. The only differences to RISE are the inclusion of timers and the methods to update the timing model.

4 Results

The fact the run time is quadratic in interval length r is easily verified through experimentation. Figure 1 shows the runtime against the interval length for a single datset from the UCR archive (UWaveGestureLibraryAll). The trend is clearly quadratic. We can also assess how well we can estimate runtime

Algorithm 2 Build c-RISE(Training data *train*, number of classifiers r, minimum interval length *minLen*)

Let $\mathbf{F} \leftarrow < F_1 \dots F_r >$ be the trees in the forest.
Let m be the length of series in *train*
$wholeSeriesFeatures \leftarrow$ getSpectralFeatures(*train*)
buildRandomTreeClassifier($\mathbf{F}_1$,*wholeSeriesFeatures*)
startForestTimer()
for $i \leftarrow 2$ to r AND queryForestTimer() **do**
    $max \leftarrow$ findMaxIntervalLength()
    startTreeTimer()
    $startPos \leftarrow$ randBetween$(1, m - max)$
    $endPos \leftarrow$ randBetween$(startPos + max)$
    $endPos \leftarrow$ nearestPowerOfTwo$(startPos, endPos, m)$
    $interval \leftarrow$ removeAttributesOutsideOfRange$(train, startPos, endPos)$
    $intervalFeatures \leftarrow$ getSpectralFeatures(*interval*)
    buildRandomTreeClassifier($\mathbf{F}_i$,*intervalFeatures*)
    $y \leftarrow$ queryTreeTimer()
    updateAdaptiveModel$(endPos - startPos, y)$

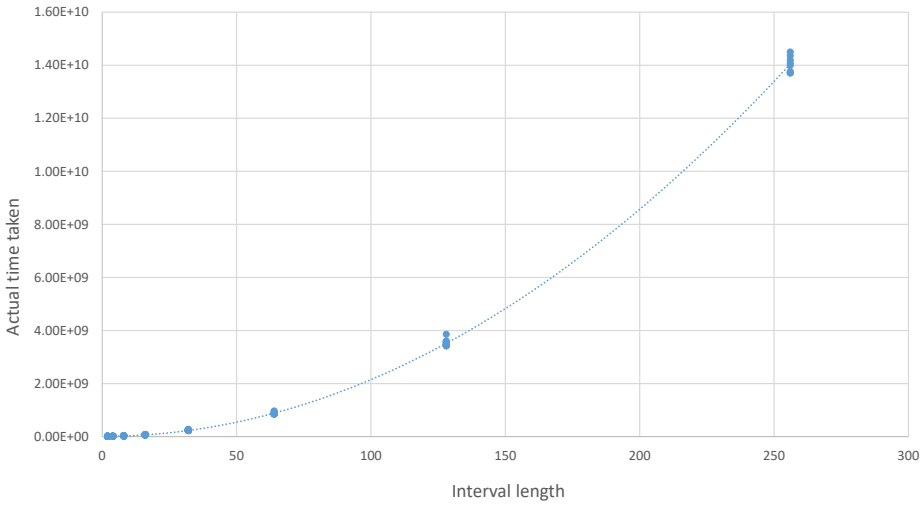


Fig. 1. Graph showing the runtime from the UWaveGestureLibraryAll dataset, time over length for $\{2, 4, 8, 16, \dots, 256\}$ trees

through experimentation. For the UWaveGestureLibraryAll dataset used to create Figure 1, the average prediction error of the trees built to model was just 0.06%.

We demonstrate the use of c-RISE with a case study. The automatic identification of birds by their call would allow for widespread monitoring of population dynamics by inexpensive sensor nets. It is also a problem for which spectral approaches are appropriate. We formulated a bird call classification problem of our bird species: Canadian Goose; Greylag Goose; Pink-footed Goose; and White-faced Whistling Duck for audio calls obtained from Xeno Canto site[1]. The recordings ranged in length from a few seconds to roughly 6 minutes, sampled at 44,100 HZ. To demonstrate how RISE works, we desire problems that may contain only a partial signal. We also want a problem that is non-trivial. Hence, the instances were all truncated to 30,000 attributes (less than one second of sound). We have 20 instances of each class. The dataset is called DuckDuckGeese and is available for download from the TSC website.

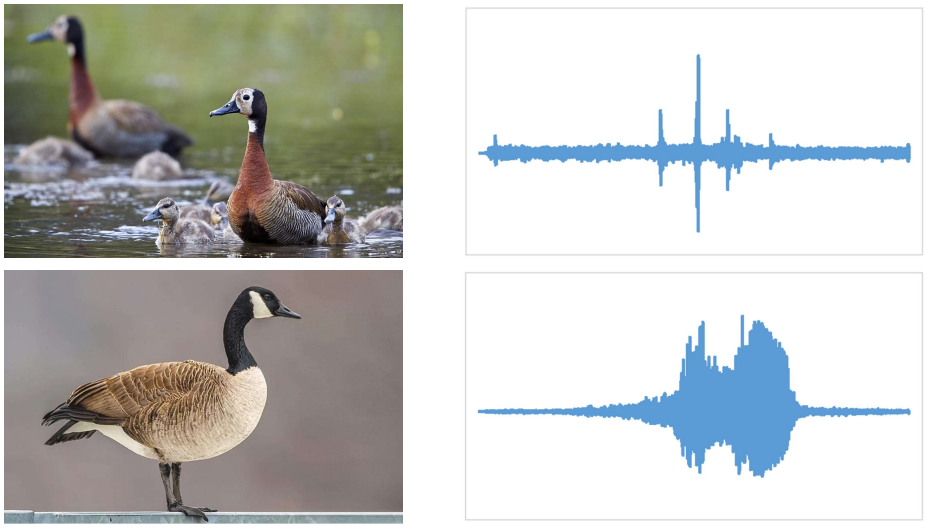


Fig. 2. Two pairs of images showing the White-faced whistling duck (top) and the Canadian goose (bottom) with example complete waveforms.

Table 1 shows the results for five classifiers on the DuckDuckGeese problem. We include three time domain classifiers to highlight the fact that spectral approaches are better for this type of data. Both RISE and c-RISE are significantly better than all three time domain classifiers (using a paired t-test). RISE takes approximately 3.5 hours to build. c-RISE has significantly lower accuracy than RISE, demonstrating the likely trade off in accuracy from contracting. The

[1] https://www.xeno-canto.org/

Table 1. Results of five classifiers on the DuckDuckGeese problem averaged over 30 resamples.

Classifier	Accuracy	Standard deviation
1-NN Euclidean Distance	31%	4.3%
1-NN Dynamic Time Warping	36.75%	6.6%
Time Series Forest	28.25%	6.1%
RISE	46.83 %	7.8%
c-RISE (30 minute contract)	42.17%	9.3%

low overall accuracy of RISE and c-RISE is probably due to our formatting decisions. We took a very small segment of the signal (less than a second) and Figure 2 demonstrates how the first part is less likely to discriminate. We intend reformatting this data to make it more useful. Another option within each ensemble member would be to randomise downsampling either within a selected interval or for the whole signal. This would allow us to build bigger ensembles and might even improve performance for audio classification in particular.

5 Conclusions

The ability to be able to predict the run time of a classifier is clearly very desirable, particularly with the increased prevalence of massive data sets. Approaches to time series classification such as HIVE-COTE are computationally intensive, when used with the default settings. However, much of that computation is not necessarily required. For example, the default shapelet transform used in COTE [3] does a full enumeration of the space of all shapelets. This takes a very long time, and is completely unnecessary (sampling a tiny fraction of the search space gives equivalent results) [4]. Our research agenda is to configure the component classifiers of COTE so that the user can specify an allowed amount of time, and to persist the classifier so that if required, more resources can be expended on building it later. We describe our first attempt at configuring one of the HIVE-COTE components, RISE, in this way. We adaptively calculate the maximum interval that can be taken with the time remaining, then constrain RISE to this interval size. We evaluate this contract classifier, c-RISE, on a new case study on classifying bird calls. The results show the benefit of spectral methods on audio problems, and highlight the tradeoff between accuracy and speed. We wish to develop c-RISE further to help close the gap in accuracy and test on much larger time series problems. Two future innovations are planned. Firstly, we will assess the usefulness of combining interval selection with downsampling. Our rationale for taking intervals is that the useful area of the signal may be padded with confounding noise. However, even if we find the correct interval, the data may be sampled at much higher a rate than is required. The randomised nature of RISE makes it easy to try combining both interval and downsampling approaches.

Acknowledgement This work is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) [grant number EP/M015807/1]. The experiments were carried out on the High Performance Computing Cluster supported by the Research and Specialist Computing Support service at the University of East Anglia and using a Titan X Pascal donated by the NVIDIA Corporation.

References

1. A. Bagnall and G. Janacek. A run length transformation for discriminating between auto regressive time series. *Journal of Classification*, 31:154–178, 2014.
2. A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.
3. A. Bagnall, J. Lines, J. Hills, and A. Bostrom. Time-series classification with COTE: The collective of transformation-based ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 27:2522–2535, 2015.
4. A. Bostrom and A. Bagnall. Binary shapelet transform for multiclass time series classification. *Transactions on Large-Scale Data and Knowledge Centered Systems XXXII: Special Issue on Big Data Analytics and Knowledge Discovery*, pages 24–46, 2017.
5. L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
6. J. Caiado, N. Crato, and D. Pena. A periodogram-based metric for time series classification. *Computational Statistics and Data Analysis*, 50:2668–2684, 2006.
7. H. Deng, G. Runger, E. Tuv, and M. Vladimir. A time series forest for classification and feature extraction. *Information Sciences*, 239:142–153, 2013.
8. J. Lines, S. Taylor, and A. Bagnall. Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles. *ACM Transactions on Knowledge Discovery from Data*, 12(5), 2018.

On Setting Parameters with Elastic Distance Measures for Time Series Classification: A Practical Case Study with Dynamic Time Warping

Jason Lines

University of East Anglia, Norwich, United Kingdom
j.lines@uea.ac.uk

Abstract. This work seeks to investigate a simple method for speeding up the build time of elastic nearest neighbour classifiers for time series classification. Specifically, Dynamic Time Warping with warping window set through cross-validation (DTWCV) is used as the example. It is shown that, when DTWCV is given 100 possible model selections in training, it does not uniformly use the whole search space when training models for classification over the UCR/UEA datasets. We conduct classification and timing experiments to show that through a simple and informed restriction to the parameter search, test accuracies that are not significantly different to using the full parameter search can be achieved while taking approximately only 13% of the time.

Keywords: time series · classification · elastic distance measures

1 Introduction

The current state of the art for time series classification (TSC) is HIVE-COTE: The Hierarchical Vote Collective of Transformation-based Ensembles [11]. HIVE-COTE is a meta-ensemble that combines predictions of five constituent ensembles built on different feature spaces. These include shapelet, interval, frequency, and spectral features, in addition to time domain classification using the Elastic Ensemble [9] (EE). Individually, each ensemble performed strongly in a recent experimental evaluation of leading TSC algorithms [1], but HIVE-COTE outperformed each individual ensemble through dynamically combining predictions of each. However, the key strength of HIVE-COTE is also the cause of one of its weaknesses. Building classifiers in multiple domains is time consuming and relatively slow. HIVE-COTE is very effective for offline problems, but the time complexity may be prohibitive for use cases where real-time or on-chip classification is necessary. The key to significantly reducing the run time of HIVE-COTE is through addressing the run time of its constituents. One of the bottlenecks within the algorithm, and other TSC research, is the training effort that is required to set parameters of elastic distance measures for use with nearest neighbour (NN) classifiers.

We present a case study that focuses on accelerating parameter optimisation for elastic distance measures in TSC research. This case study initially focuses on a single measure, but the findings have implications for EE, HIVE-COTE, and NN classification for TSC as a whole. For context, EE currently uses a combination of 11 NN classifiers with various elastic distance measures. 8 of these require parameter tuning, where each is given 100 model selections to be evaluated using leave-one-out cross-validation (LOOCV) in training. This is clearly a key area of complexity within EE, and thus a bottleneck within HIVE-COTE. While measure-specific speedups exist, our hypothesis is that parameter selection for elastic NN classifiers can be significantly accelerated without loss of test accuracy through a simple, informed restriction of the parameter search space.

We investigate this hypothesis using one of the most widely-used elastic distance measure in TSC: Dynamic Time Warping (DTW). DTW with a 1-NN classifier has long been used as a comparative benchmark in TSC research and is also a constituent classifier within EE. The strength of DTW is that it allows for series alignment in the presence of local phase-shift in the time axis. While it is possible to consider full warping of series for alignment, a *warping window* is typically used to restrict alignments between series. Using a warping window in this manner has been shown to significantly improve test accuracy. As such, EE includes DTWCV, a 1-NN classifier using DTW that is given 100 model selections, where each option corresponds to a window width from 1% to 100% and training LOOCV is used to select the optimal width for test classification. In this work, we leverage the results from [1, 9, 11] to investigate how DTW uses these 100 model selections in practice over 100 resamples of 85 UCR/UEA TSC problems. We perform a decomposition of existing results to determine which values are selected and show that the choices are not uniform across the search space. We use this knowledge to restrict the parameter space, and conduct new experiments to demonstrate that almost identical accuracy can be achieved across the 100 resamples of the 85 datasets through using only 13% of the parameter space. Finally, we conduct novel timing experiments using the full search and restricted search spaces to demonstrate the real-world benefit of restricting the search space in this manner.

2 Background and Related Work

For many years DTW 1-NN was considered the gold standard to TSC. Specifically, DTW-1NN with a warping window set though LOOCV on the training data (DTWCV) has been shown to significantly outperform a full-window approach [9, 1, 7, 15, 5, 3]. Further effort has been placed on DTW-specific speedups, such as GPU acceleration [12] and lower bounds [8]. A recent promising approach in [14] considers bounds across parameter options to further accelerate DTW.

Alternative TSC approaches have recently been shown to significantly outperform DTW however. The work in [9] investigated various alternative elastic distance measures to DTW. The findings showed that while no single measure significantly outperformed DTW, an Elastic Ensemble (EE) that combined NN

classifiers built with various elastic measures (including DTWCV) significantly outperformed DTWCV and all individual components. EE consists of 11 constituent 1-NN classifiers. Each is coupled with an elastic distance measure: 3 without parameters (Euclidean, DTW, Derivative DTW (DDTW)) and eight with parameters (DTW and DDTW with warping windows, Weighted DTW and DDTW, Longest Common Subsequence, Edit Distance with Real Penalty, Move-Split-Merge, Time Warp Edit). The parameters for each are set using LOOCV on the training data.

EE was later incorporated into Flat-COTE [2], which combined more than 30 classifiers built on different representations of the data. Flat-COTE was shown to be the state of the art for TSC in [1], and was a precursor to the current state of the art: HIVE-COTE [11]. HIVE-COTE is a meta-ensemble that contains constituent ensembles built on five different data representations. The predictions of each are combined for test prediction using a proportional voting scheme that is guided by weights learned in training. The five constituent ensembles are: the Elastic Ensemble [9] (EE), an ensemble built on shapelet-transformed data [4], a Bag-of-SFA-Symbols ensemble [13], Time Series Forest [6], and a random interval spectral ensemble [10].

This work specifically focuses on the training effort of building NN classifiers using elastic distance measures. While of interest for NN classifiers using single elastic distance measures, such as DTWCV, this case study has specific implications for EE. Each constituent NN classifier within EE is currently given 100 model selections to set distance measure parameters in training; this is a key source of complexity in the ensemble. We believe that there is large amount of redundancy in this parameter search and that the training time of EE could be significantly reduced by restricting the allowed parameter space for each constituent without losing test accuracy. Due to its popularity in the literature, we use DTW as the case study to investigate how parameters are selected in training across various datasets.

3 Case Study: DTW

For the following case study we use a combination of published results and novel experiments. First, we leverage from the extensive existing results of DTWCV from [1], which includes the LOOCV-selected parameters and associated test accuracies for 100 resamples of the 85 UCR/UEA datasets[1]. We provide an overview and discussion of the parameter selections, and make recommendations on the values that could be used. We then conduct new experiments with these settings and summarise differences in test accuracies and timing results.

We note that while we have selected DTW for this case study due to its popularity and ubiquitous use in the literature, the findings of this case study have implications for all EE constituents. Therefore we do not include DTW-specific enhancements from Section 2 as we do not wish to confound the impact

[1] Data available from: www.timeseriesclassification.com

of the adjustments we make in the following work. Further, EE, and each of its constituents (including DTWCV), naturally lend themselves to parallel execution of multiple model selections simultaneously. Some of the most promising speedups use results from initial parameter evaluations to inform subsequent evaluations. This dependency adds extra complexity when interpreting results so it is more transparent to consider this case study in isolation. However, we note that there is potential to incorporate additional enhancements to recommendations made from this case study for even greater benefit in future work.

3.1 Dynamic Time Warping with 100 Model Selections

DTWCV is given 100 model selections within EE, where each option corresponds to a different warping window width (relative to the full series length) in increments of 1%. In [9], experiments with EE were conducted over the default train/test splits of the 85 UCR/UEA datasets. We have deconstructed these results to uncover which window was selected for each dataset. A histogram of these window selections is presented in Figure 1.

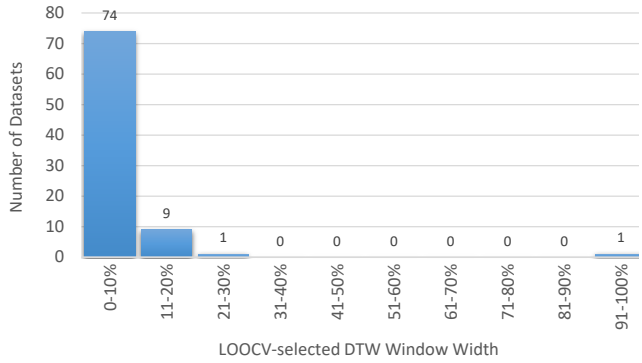


Fig. 1: A histogram of the window widths used by DTWCV over the default train/test splits of the 85 UCR/UEA datasets

The findings were somewhat surprising; DTWCV used a warping width of less than 10% for 71 datasets, and no more than 20% for 83. Only 2 datasets required windows in the remaining 80% of the parameter space: MedicalImages (21%) and LargeKitchenAppliances (94%). This initial finding suggests that it may be possible to significantly constrain the parameter search for DTWCV. To verify this, we repeated the investigation using results from [1] over 100 resamples of the same 85 datasets. The histogram in Figure 2 summarises these 8,500 window selections.

The results in Figure 2 are consistent with the previous observations; almost 95% of the window widths found in training by DTWCV in [1] were no greater than 20%. This poses the question: would restricting the parameter search in training significantly affect test classification accuracy?

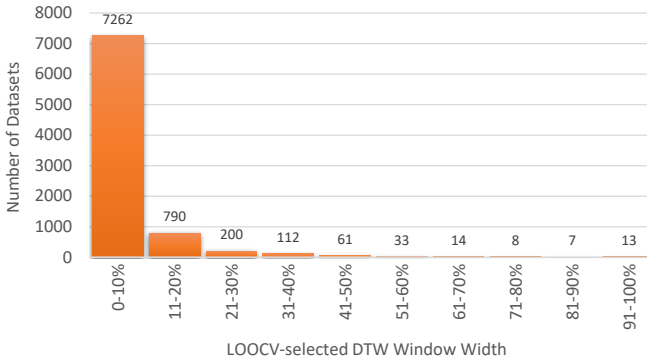


Fig. 2: A histogram of the window widths used by DTWCV over the 100 resamples of 85 UCR/UEA datasets

3.2 DTW 1NN with Restricted Parameter Search

We design an experiment to investigate the effect of restricting parameter options by building new DTWCV classifiers on 30 resamples of the 85 UCR/UEA datasets. We restrict the parameter options of each DTWCV as follows. First, informed by Section 3.1, we allow the classifiers to search over the first 20% of window sizes in training. Second, we also include the last 10% of the possible window sizes. This addition is to ensure that full warping is still available if necessary. Finally, we also note that the step-size of 1% between window sizes is very small. We do not believe that a significant difference in accuracy would be caused by using a window that is only 1% away from the optimal window. Therefore we further reduce the search space by incrementing potential window sizes in multiples of 3. The final possible window sizes that each DTWCV classifier is allowed to evaluate in training are: 0% (equivalent to Euclidean Distance), 3%, 6%, 9%, 12%, 15%, 18%, 21%, 88%, 91%, 94%, 97%, 100%. We include 21% and 88% to ensure full coverage of the first 20% and last 10% of the search space while using multiples of 3. We note that these values are somewhat arbitrary; they are based on informal findings from Section 3.1 and assumptions that we have made. They serve as an initial investigation and have not been cherry-picked or optimised, so further refinement of these values would likely yield superior results in the future. The full results are omitted for brevity; a summary is presented in Table 1 and a full listing can be accessed online[2].

Table 1 demonstrates that there is very little difference in test performance between the training strategies. The full parameter search is only 0.06% more accurate on average and the difference in rank is also marginal; the full search wins on 43 datasets with an average rank of 1.47 and the restricted search wins on 38 with a rank of 1.53. There is no significant difference according to a t-test or Wilcoxon signed-rank test. Therefore we can conclude that using a restricted parameter search did not significantly reduce test accuracy in this experiment.

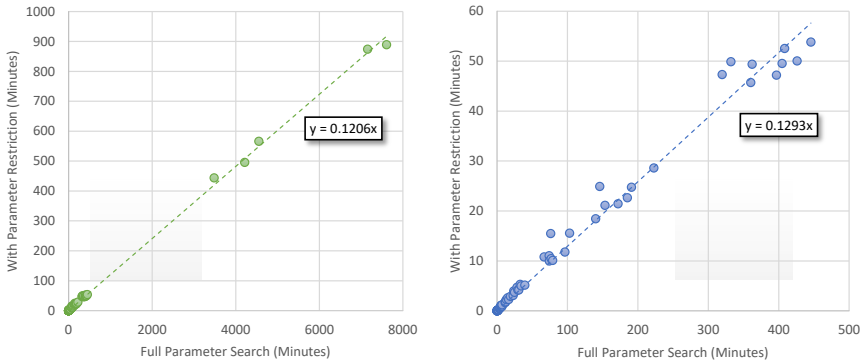
[2] Results by Dataset: https://goo.gl/oj7aJW

	Full Parameters	Restricted Parameters
Average Accuracy	77.61% (15.55)	77.55% (15.56)
Average Rank	1.47	1.53
Wins	43	38

Table 1: The results of DTWCV using 100 model selections versus a limited search space of 13 parameters. Results are averaged over 30 resamples of 85 datasets and standard deviations over the datasets are given in brackets.

3.3 Timing Comparison

We now demonstrate the utility of this restriction by conducting a timing experiment. We build DTWCV classifiers using both the full 100 parameters and the restricted space of 13 parameters over the 85 UEA/UCR datasets. The equipment used was the High Performance Computing Cluster at the University of East Anglia. We distributed experiments across an array of Intel Haswell cores (2.6GHz) and repeated timing experiments over 10 resamples of the data, taking an average for each dataset. Each problem was given a maximum of 7 days of computation time. The results of the timing experiment are summarised in the scatter plots in Figure 3.



(a) All Datasets* (b) 77 Fastest Datasets

Fig. 3: Timing results of DTCV over the UCR/UEA datasets. Each point represents window optimisation of a dataset (in minutes, averaged over 10 resamples). The plot in (a) presents results over all datasets. Due to the large difference in scale for a minority of slow problems, it is repeated in (b) for clarity using the 77 faster problems. The speedup demonstrated in both plots remains consistent.

Figure 3 demonstrates a substantial difference in the time to build DTWCV with 100 and 13 parameters. This is to be expected; it would be assumed that evaluating 13% of the parameter space would take approximately 13% of the full time. This is confirmed in Figure 3a, which includes 82 of the datasets. FordA,

FordB, and HandOutlines each exceeded the 7 day limit when using the full parameter search space and are not included in Figure 3. However, all three problems finished within 3 days of computation time using the restricted search, further demonstrating the utility of this simple speedup adjustment. While these results are as expected, we believe they will be of interest to practitioners in the field as wall-clock timing experiments are not prevalent in the TSC literature.

These findings are very promising given the simple nature of this training adjustment. However, we caution that these results should be interpreted with care. The parameter restrictions are clear and simple, but are based on reflection of previous results. The UCR/UEA repository contains a diverse range of problems; if these are representative of all TSC problems then parameter observations are valid for all problems. However, if they are not fully-representative, it is possible new problems may benefit from parameter options outside of the current range. To that end, our guidance is that a full parameter search makes sense when resources are available, but adjustments such as those used in this work are clearly beneficial where resources are restricted or rapid prototyping is sufficient.

4 Conclusions and Future Work

This work has demonstrated a very simple and transparent method for accelerating parameter optimisation of elastic distance measures for nearest neighbour time series classification. A case study has been presented using Dynamic Time Warping 1-NN with warping set through cross-validation on the training data. Through enforcing a simple restriction of the parameter search space, the classifier was able to achieve almost identical test accuracies over 30 resamples of the 85 UCR/UEA datasets. This was by only considering 13% of the possible parameter space, achieving approximately a 7-8 times speedup of window optimisation without any significant loss in accuracy. It should be noted that this initial study only used a simple criteria for parameter restriction, and further work will determine whether there is potential to enforce additional restrictions to the parameter search space without loss of test accuracy.

These findings have implications for other elastic distance measures used with NN classifiers in TSC, as well as ensemble approaches that include them. It is anticipated that similar parameter restrictions for other elastic distance measures would yield comparable speedups without loss of test accuracy. Further work is required to verify this however, but such findings would have significant implications for EE and HIVE-COTE. Such speed-ups would enable ensemble approaches such as these to become more feasible for larger problems where restrictions on training times are important. Further, there is also scope to couple parameter search restrictions with measure-specific speedups, such as lower-bounds and GPU acceleration, which would make great progress in moving state-of-the-art TSC algorithms towards facilitating real-time or on-chip classification for a wider range of problems in the future.

Acknowledgements

This work is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) [EP/M015807/1]. Experiments were carried out on the High Performance Computing Cluster supported by the Research and Specialist Computing Support service at the University of East Anglia, and this research is also supported by a Titan XP generously donated by the NVIDIA Corporation.

References

1. Bagnall, A., Lines, J., Bostrom, A., Large, J., Keogh, E.: The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery **31**(3), 606–660 (2017)
2. Bagnall, A., Lines, J., Hills, J., Bostrom, A.: Time-series classification with COTE: The collective of transformation-based ensembles. IEEE Transactions on Knowledge and Data Engineering **27**, 2522–2535 (2015)
3. Bagnall, A., Lines, J.: An experimental evaluation of nearest neighbour time series classification. arXiv preprint arXiv:1406.4757 (2014)
4. Bostrom, A., Bagnall, A.: Binary shapelet transform for multiclass time series classification. Trans. on Large-Scale Data and Knowledge Centered Systems XXXII: Special Issue on Big Data Analytics and Knowledge Discovery pp. 24–46 (2017)
5. Dau, H.A., Silva, D.F., Petitjean, F., Forestier, G., Bagnall, A., Mueen, A., Keogh, E.: Optimizing dynamic time warpings window width for time series data mining applications. Data Mining and Knowledge Discovery pp. 1–47 (2018)
6. Deng, H., Runger, G., Tuv, E., Vladimir, M.: A time series forest for classification and feature extraction. Information Sciences **239**, 142–153 (2013)
7. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: Experimental comparison of representations and distance measures. In: Proc. 34th International Conference on Very Large Data Bases (2008)
8. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowledge and information systems **7**(3), 358–386 (2005)
9. Lines, J., Bagnall, A.: Time series classification with ensembles of elastic distance measures. Data Mining and Knowledge Discovery **29**, 565–592 (2015)
10. Lines, J., Taylor, S., Bagnall, A.: HIVE-COTE: The hierarchical vote collective of transformation-based ensembles for time series classification. In: Proc. IEEE International Conference on Data Mining (2016)
11. Lines, J., Taylor, S., Bagnall, A.: Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles. ACM Transactions on Knowledge Discovery from Data (2018)
12. Sart, D., Mueen, A., Najjar, W., Keogh, E., Niennattrakul, V.: Accelerating dynamic time warping subsequence search with gpus and fpgas. In: 2010 IEEE International Conference on Data Mining. pp. 1001–1006 (Dec 2010)
13. Schäfer, P.: The BOSS is concerned with time series classification in the presence of noise. Data Mining and Knowledge Discovery **29**(6), 1505–1530 (2015)
14. Tan, C.W., Herrmann, M., Forestier, G., Webb, G.I., Petitjean, F.: Efficient search of the best warping window for dynamic time warping. In: Proceedings of the 2018 SIAM International Conference on Data Mining. pp. 225–233. SIAM (2018)
15. Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., Keogh, E.: Experimental comparison of representation methods and distance measures for time series data. Data Mining and Knowledge Discovery **26**(2), 275–309 (2013)

Should we Reload Time Series Classification Performance Evaluation ? (a position paper)

Dominique Gay[1] and Vincent Lemaire[2]

[1] Laboratoire d'Informatique et de Mathématiques EA2525
Université de La Réunion, La Réunion, France
[2] Orange Labs
Lannion, France

Abstract. Since the introduction and the public availability of the UCR time series benchmark data sets, numerous Time Series Classification (TSC) methods has been designed, evaluated and compared to each others. We suggest a critical view of TSC performance evaluation protocols put in place in recent TSC literature. The main goal of this "position" paper is to stimulate discussion and reflexion about performance evaluation in TSC literature.

1 Introduction

The *need for time series data mining benchmarks* [15] has been fulfilled: with, firstly, the UCR Time Series Classification Archive [9] and then the UEA & UCR Time Series Classification Repository [5], the research community now have more than 85 data sets from various application domains to evaluate newly introduced TSC methods and to compare them to existing ones. The public availability and the wide diffusion of the benchmark data had a strong and positive impact on the research community which has been prolific in terms of publications of TSC methods; as an example, the recent experimental evaluation in [3] involves 18 recently proposed algorithms while in the same year, in 2017, two contenders, BOPF [17] and WEASEL [23] have been presented in top data mining conferences.

In most of the research papers, the experimental evaluation section starts with *"Each UCR dataset provides a train and test split set which we use unchanged to make our results comparable the prior publications"*[3]. And this is where the reflexion we suggest begins. In the following, we will consider accuracy as the measure of predictive performance since it is the most widely used in TSC literature.

[3] quoted from WEASEL paper [23].

2 Discussion & reflexion

About train/test experiment.
While in transactional data classification, resampling strategies (mainly bootstrap and stratified k-fold cross-validation) are the norm to estimate the expected performance of classifiers and to compare them [22, 11], there is a singularity concerning predictive performance evaluation in TSC literature: the vast majority of research work restrains predictive performance evaluation to a single train/test split experiment [5], also called hold-out method or test sample estimation [16].

Unless disposing of a *large and representative* data set of the application domain, a single train/test experiment is generally ineffective in providing predictive performance estimation and valuable comparison between TS classifiers without random subsamplings (i.e., repeated hold out experiments). Indeed, different samplings may lead to results with strong variations [16]. Thus, a classifier $\mathcal{C}_A$ may show better predictive performance than another classifier $\mathcal{C}_B$ just because of this particular split. And, in such cases, subsequent statistical tests based on single train/test accuracy results (such as now commonly used post-hoc Nemenyi test [12]) do not help more in comparing classifiers performance over several data sets since other train/test splits could have led to different accuracy and mean rank results and thus potentially different conclusions.

Moreover, unless disposing of tens of thousands instances, it is common to keep more instances in the training set than in the test hold out, generally around class-stratified ratio of 2/3 for training against the rest i.e., 1/3 for testing. The TSC repository [5] provides various predefined train/test split ratios, going from 1.6% to 81% of the whole data set for training set (notice that 35 of the 85 TSC data sets show a train/test split ratio below 34%). In addition to small –if not very small– training sets, the train/test splits are not always class-stratified; it results in very (too) few representative instances of some class labels (especially in multi-class problems). We are aware that, in some application domains, "labelled data is expensive to collect" [3], however, considering the whole available and *already-labelled* data, splitting for such small non-stratified training sets is also questionable. Here are some singularities that arise from some data sets drawn from the TSC archive:

- DiatomSizeReduction: a 4-class problem, with a train/test split ratio about 5% (16 instances for training, 306 for testing) and a class distribution (1,6,5,4) for training against (33,92,94,87) for testing. The class distribution is not respected from train to test set. There are relatively 73% more instances of class c_1 in the test set than in the training set.
- SonyAIRobotSurface1: a 2-class problem, with a train/test split ratio about 3.22% (20 instances for training, 601 for testing) and a class distribution (6,14) for training against (343,258) for testing. The class distribution is not respected from train to test set. There are relatively 90% more instances of class c_1 in the test set than in the training set; and 38% less instances of class c_2 in the test set than in the training set.
- ... several additional similar cases of data sets can be found in commonly used benchmark [5] : either the number of training examples is very small

w.r.t the whole available data, either there is class distribution change between training set and test set (sometimes both singularities arise).

Generally speaking, in addition to the weakness of single train/test experiments for performance evaluation, choosing to split such way, without class-stratification in small training sets, results in "unnecessary difficult" TSC problems: indeed, the obtained training set is not always representative of the whole available data set and as explained above, it could lead to class distribution change between training and test sets (also known as prior probability shift [20]). In such environments, intuitively, simple 1-Nearest Neighbor lazy learners [27] and ensemble methods that embed several 1-NN classifiers (with various distance measures or on various data representations) [2, 4, 18] still obtain "good" accuracy results since it is still possible to find a nearest neighbor of a test instance in very few training instances of a minor class. However, eager classifiers based on empirically observed per-class frequencies (such as Naïve Bayes or Decision Trees) often fail in characterizing the minor classes with very few representative instances.

If the TSC archive [5] offers a wide variety of data sets, the original train/test splits also involves hidden difficult well-known problems in the Machine Learning community: e.g., learning from few examples or in class distribution change environment. Unfortunately, averaging ranks over all data sets to produce critical difference diagram presents a risk of hiding the reasons why a particular classifier shows better performance than another. As an example [4], in Table 1, we report performance comparisons of 11 recent classifiers like in recent literature (single train/test split following by significance testing). We provide several statistical tests by integrating step by step data sets which involves smaller size of training set.

Min. size	#DB	CD	WEASEL	DTWCV	DTW	BOSS	LS	TSBF	ST	EE	CoTE	SNB	BoPF
≥ 1000	10	*4.77403*	**3.250**	8.300	8.600	**5.700**	**6.050**	**7.700**	**3.050**	**6.250**	**4.100**	**3.400**	9.600
> 500	22	*3.21865*	**2.932**	7.659	9.023	**5.705**	6.909	6.841	**4.068**	6.273	**3.636**	**4.000**	8.955
> 300	42	*2.32949*	**3.560**	7.929	8.893	5.857	6.845	6.298	**4.500**	6.060	**3.369**	**4.821**	7.869
> 200	48	*2.17903*	**3.594**	7.917	8.885	5.781	6.885	6.271	**4.625**	5.865	**3.219**	**5.010**	7.948
> 100	57	*1.99962*	**3.737**	7.868	8.860	5.474	6.623	6.237	**4.544**	5.930	**3.298**	5.535	7.895
All	85	*1.63748*	**3.847**	7.806	8.647	5.382	6.135	6.388	**4.847**	5.871	**3.412**	6.429	7.235

Table 1. Average ranks of 11 recent classifiers depending on the data sets taken into account from [5] (i.e., depending on the training set size). Accuracy results are taken from Schäfer & Leser paper on WEASEL [23]. Post-hoc Nemenyi's statistical test considering training set with size ≥ 1000 (only 10 data sets), then > 500 (only 22 data sets), ..., until considering All 85 data sets from [5]. Underlined rank is the best per line and bold results on the same line indicates that there is no statistically significant difference of performance between bold results according to Nemenyi's test, considering the current benchmark datasets.

[4] We did not integrate recently introduced HIVE-COTE [19] accuracy results since they are available under 100 resamples protocol.

WEASEL and ST [14, 7] score the highest mean ranks when considering data sets with training set size greater than 500 while CoTE takes advantage when adding data with smaller training size. We also observe that the mean rank of ST increases as we consider more and more data sets with smaller training size. Notice that, it has to be balanced against the fact that as the number of data sets decreases the critical difference (CD) value increases, making more difficult to state significant differences of performance between contenders. Another illustrative example is about SNB [8] (an improved Naïve Bayes classifier benefiting from multiple representations) which is competitive with WEASEL, CoTE, BOSS and ST when not considering too small (< 200) training set size – confirming the importance of the size of the training set on the predictive performance of some classifiers. Aside from [27], we may regret the lack of experimental studies about the learning curves [21, 24] of TSC algorithms.

About resampling strategies.
As far as we know, very few attempts of resampling strategies for TSC performance evaluation have been led, e.g., :

- Grabocka et al. [13] provides some results by 5-folds cross validation on 35 UCR data sets. It allows to identify easy data sets in the TSC archive [9]. Indeed, a default SVM classifier with polynomial kernel scores above 95% accuracy (often near perfect) on 18 data sets.
- Wang et al. [27], focusing on 1-NN with various distance measures, provides k-folds stratified cross-validation results. However, the k varies from 2 to 30 depending on the benchmark data set and the cross-validation method at use is unconventional: when splitting the data set T into k folds, the model is learnt on fold T_k and tested on $T \setminus T_k$ –while conventional k-folds cross validation being the opposite: learning the model on $T \setminus T_k$ and testing on T_k. This unconventional cross-validation leads to the same problems explained above (with 1/30, i.e., 3% of the whole data set used for training).
  In the same paper, the authors also noticed the importance of the effect of the training data set size on 1-NN classifier accuracy : DTW-1-NN is better than ED-1-NN with small training data set, but providing that we have enough training instances (a few hundred/thousand depending on the simulated data set), the two classifiers show similar predictive performance.
- Bagnall et al. [3] performs 100 resampling experiments on each of the 85 TSC data sets –followed by Nemenyi's statistical post-hoc test. However, the multiple resamplings fit the original size of train/test split provided by [5] – which leads to the same problem raised above about training size and class distribution change. This resampling strategy gives, all the same, a better idea of the performance of recent classifiers on data with *various* sizes of training sets, although the train/test split is still questionable. Indeed, if instances from original test set are authorized to be in training set due

to resampling, why not use 10-CV or 10×10 CV, as in transactional data classification literature[5] ?

The cross-validation (CV) method is not unknown to the TSC community; indeed, some algorithms, like e.g., WEASEL or DTW-CV used cross-validation on training set to set hyper-parameters (even if the training set is very small), then a single train/test split is performed to evaluate the performance of the "best hyper-parametered" model on a single hold out test set. Again, why not use CV method to evaluate and compare TSC algorithms ?

About (repeated) 10-folds CV, statistical tests and beyond.
While 10-folds CV with subsequent statistical tests [12] is now the gold standard for predictive performance evaluation and comparisons between classifiers on transactional benchmark data sets, recently, Vanwinckelen & Blockeel [25, 26] warns the Machine Learning community about pitfalls hidden in such comparisons. The take away messages are:

- *"Our experiments show that when using cross-validation for choosing between two models, the best performing model is not always chosen".*
- *"This discussion leads us to question the usefulness of statistical testing in the context of evaluating predictive models with cross-validation".*

On the other hand, after almost a decade of the *"10-CV + statistical tests"* combination [12] to evaluate learner's predictive performance, J. Demsar et al. [6, 10] "*discourage the use of frequentist null hypothesis significance tests (NHST) in machine learning and, in particular, for comparison of the performance of classifiers*" and encourages the community to embrace Bayesian analysis using 10×10-CV for comparing classifiers. This is perhaps a change point for performance evaluation habits in the Machine Learning community. Notice that Bayesian analysis of performance is more conservative than NHST; that is, it is "more difficult" for a classifier $\mathcal{C}_A$ to be better than a classifier $\mathcal{C}_B$, considering Bayesian analysis.

About the evaluation measure.
As recalled in the introduction, the vast majority of recently proposed TSC algorithms are evaluated and compared with regards to the accuracy measure, i.e., the number of correctly classified time series. Accuracy measure is suitable for roughly balanced 2-class data sets. However, for unbalanced and/or multiclass data sets, accuracy measure is inappropriate for evaluation since high accuracy results due to a bias towards the majority class could hide severely bad predictive performance on the minor class or on other classes in multiclass settings. Often, ROC or Precision/Recall curve analysis or lift curve and cumulative gain charts are prefered for unbalanced settings.

The TSC archive [5] contains some 2-class unbalanced problems, e.g., Earthquakes, ToeSegmentation2 and Wafer with respectively 20.2%, 25.3% and 10.6%

[5] However, even with k-folds CV, a particular attention must be given to the setting of k for the 17 small data sets, with less than 200 instances, from the TSC archive.

unbalanced ratio (i.e., the proportion of the minor class). The archive also contains many multiclass problems with severe unbalanced ratios, e.g., ECG5000 and Worms with respectively 0.5% and 9.7% unbalanced ratio. Learning in the presence of class imbalance or in multiclass settings is still an ongoing machine learning research topic [1]. Again, the presence of such data sets in the benchmark repository, when averaging ranks of classifiers based on accuracy results, could lead to flawed conclusions on the performance evaluation.

3 Conclusion

The still ongoing public release of benchmark TSC data sets to the data mining community through the UCR & UEA repository has certainly been the best catalyst for the development of new TSC algorithms and methods. In this discussion paper, we briefly review the habitual protocols at use in TSC algorithms performance evaluation, discuss the pros and cons of experimental protocols and try to warn the community about the pitfalls and hidden problems of actual performance evaluation protocols in TSC literature. We agree that the core of this discussion paper needs more in-depth development and experimental arguments but we believe that interesting discussions on this important topic deserve to be launched and continued during the 3^{rd} ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data.

Acknowledgments

We would like to thank the anonymous reviewers for joining the discussion with valuable comments and arguments.

References

1. Proceedings of the IJCAI 2017 Workshop on Learning in the Presence of Class Imbalance and Concept Drift (LPCICD'17) (2017)
2. Bagnall, A., Davis, L.M., Hills, J., Lines, J.: Transformation based ensembles for time series classification. In: Proceedings of the Twelfth SIAM International Conference on Data Mining, Anaheim, California, USA, April 26-28, 2012. pp. 307–318 (2012)
3. Bagnall, A., Lines, J., Bostrom, A., Large, J., Keogh, E.J.: The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Min. Knowl. Discov. 31(3), 606–660 (2017)
4. Bagnall, A., Lines, J., Hills, J., Bostrom, A.: Time-series classification with COTE: the collective of transformation-based ensembles. IEEE Trans. Knowl. Data Eng. 27(9), 2522–2535 (2015)
5. Bagnall, A., Lines, J., Vickers, W., Keogh, E.: The UEA & UCR time series classification repository, `www.http://timeseriesclassification.com`

6. Benavoli, A., Corani, G., Demsar, J., Zaffalon, M.: Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis. Journal of Machine Learning Research 18, 77:1–77:36 (2017)
7. Bostrom, A., Bagnall, A.: Binary shapelet transform for multiclass time series classification. In: Big Data Analytics and Knowledge Discovery - 17th International Conference, DaWaK 2015, Valencia, Spain, September 1-4, 2015, Proceedings. pp. 257–269 (2015)
8. Boullé, M.: Towards automatic feature construction for supervised classification. In: Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part I. pp. 181–196 (2014)
9. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The UCR time series classification archive (July 2015), `www.cs.ucr.edu/~eamonn/time_series_data/`
10. Corani, G., Benavoli, A., Demsar, J., Mangili, F., Zaffalon, M.: Statistical comparison of classifiers through bayesian hierarchical modelling. Machine Learning 106(11), 1817–1837 (2017)
11. Delgado, M.F., Cernadas, E., Barro, S., Amorim, D.G.: Do we need hundreds of classifiers to solve real world classification problems? Journal of Machine Learning Research 15(1), 3133–3181 (2014)
12. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (2006)
13. Grabocka, J., Nanopoulos, A., Schmidt-Thieme, L.: Invariant time-series classification. In: Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II. pp. 725–740 (2012)
14. Hills, J., Lines, J., Baranauskas, E., Mapp, J., Bagnall, A.: Classification of time series by shapelet transformation. Data Min. Knowl. Discov. 28(4), 851–881 (2014)
15. Keogh, E.J., Kasetty, S.: On the need for time series data mining benchmarks: A survey and empirical demonstration. Data Min. Knowl. Discov. 7(4), 349–371 (2003)
16. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes. pp. 1137–1145 (1995)
17. Li, X., Lin, J.: Linear time complexity time series classification with bag-of-pattern-features. In: 2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18-21, 2017. pp. 277–286 (2017)
18. Lines, J., Bagnall, A.: Time series classification with ensembles of elastic distance measures. Data Min. Knowl. Discov. 29(3), 565–592 (2015)
19. Lines, J., Taylor, S., Bagnall, A.J.: HIVE-COTE: the hierarchical vote collective of transformation-based ensembles for time series classification. In: IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain. pp. 1041–1046 (2016)
20. Moreno-Torres, J.G., Raeder, T., Alaíz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. Pattern Recognition 45(1), 521–530 (2012)
21. Perlich, C., Provost, F.J., Simonoff, J.S.: Tree induction vs. logistic regression: A learning-curve analysis. Journal of Machine Learning Research 4, 211–255 (2003)
22. Salzberg, S.: On comparing classifiers: Pitfalls to avoid and a recommended approach. Data Mining & Knowledge Discovery 1(3), 317–328 (1997)

23. Schäfer, P., Leser, U.: Fast and accurate time series classification with WEASEL. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017. pp. 637–646 (2017)
24. Ting, K.M., Washio, T., Wells, J.R., Aryal, S.: Defying the gravity of learning curve: a characteristic of nearest neighbour anomaly detectors. Machine Learning 106(1), 55–91 (2017)
25. Vanwinckelen, G., Blockeel, H.: On estimating model accuracy with repeated cross-validation. In: Proceedings BeneLearn'2012 (2012)
26. Vanwinckelen, G., Blockeel, H.: Look before you leap: Some insights into learner evaluation with cross-validation. In: 1st ECML/PKDD Workshop on Statistically Sound Data Mining, SSDM 2014, held at ECML/PKDD 2014, Nancy, France, September 15, 2014. pp. 3–20 (2014)
27. Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., Keogh, E.J.: Experimental comparison of representation methods and distance measures for time series data. Data Min. Knowl. Discov. 26(2), 275–309 (2013)

Triclustering based outlier-shape score for time series in a fraud detection platform

Pierre Lejeail[1,2], Vincent Lemaire[1], Antoine Cornuéjols[2], Adam Ouorou[1]

[1] Orange Labs, 2 Avenue Pierre Marzin, 22300 Lannion, France
[2] AgroParisTech, 16 rue Claude Bernard, 75005 Paris, France

Abstract. This paper presents a triclustering based outlier-shape score for time series in the context of a fraud detection platform for wholesale traffic for a telecommunications carrier. We propose to use triclustering as an exploration module for outlier shape detection using whole time series. Three main steps compose this approach: (1) projection of data in a new space of time series related features (e.g. derivative), (2) estimation of the density of known normal data using a triclustering method (3) computation of an outlierness score quantifying the distance to the estimator from step (2). We conduct an evaluation of the methodology by focusing on its ability to separate data from different classes. Our preliminary results to assess this approach are very encouraging.

1 Introduction - Context

Telecommunications companies in different countries use a variety of international telecoms routes to send traffic to each other. In a "wholesale market", telecom carriers can obtain traffic to make up a shortfall, or send traffic on other routes, by trading with other carriers in the wholesale or carrier-to-carrier market. Minutes exchanges allow carriers to buy and sell call terminations (dispatch of a call to its destination). Prices in the wholesale market can change on a daily or weekly basis. A carrier will look for least cost routing function to optimize its trading on the wholesale market.

The quality of routes on the wholesale market can also vary, as the traffic may be going on an illegal route.

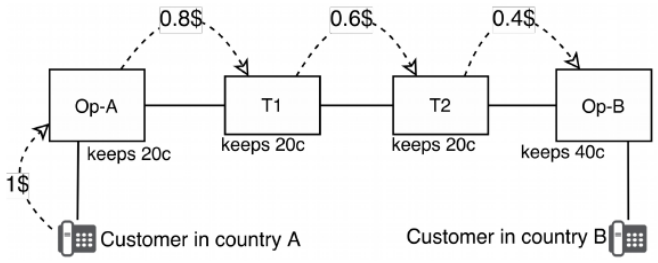


Fig. 1. A value chain providing a call termination between two users

A value chain (as the example in Fig. 1) exists between operators that provide the connection between two customers. But this value chain could be broken if a fraudster finds a way to generate communication without paying. For the last decades, fraud has been a growing concern in the telecommunication industry. In a 2017 survey [1], the CFCA[3] estimates the annual global fraud loss of $30 billion (USD). Therefore, detecting and preventing, when possible, is essential in this domain. Regarding wholesale market, a list of fraud is known [2] by the operators and fraud detection platforms already exist.

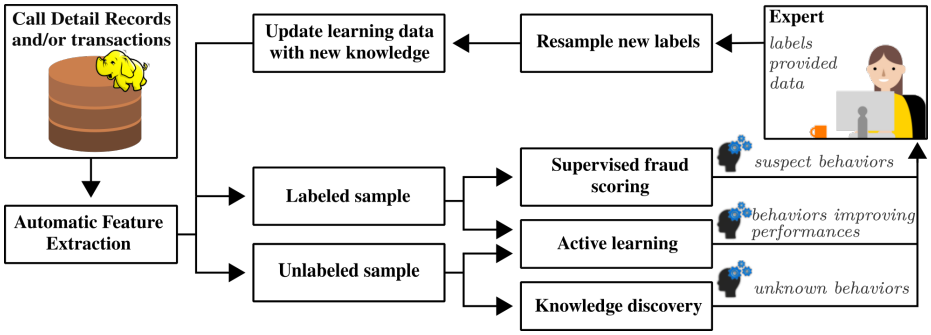


Fig. 2. The Orange platform for fraud detection

One of these platforms, shown in in Fig. 2, has been realized by Orange (as a wholesale operator). This platform contains modules which **exploit** the information given by the expert (a scoring module, for example, which is a classifier), others **explore** the data to interact with the expert(s) by finding new patterns, including (but not only) malevolent ones. This goal is achieved by knowledge discovery and active learning techniques. Those exploration modules are responsible for adapting to the constant evolution of fraudster's behaviors under (in the case of this platform) the constraint of the limited time that experts can afford to spend in this exploration. This platform share similarities with the one presented by Veeramachaneni et al. in [3]. Both platforms combine a supervised model for predictions with unsupervised models for exploration of unknown pattern, and takes into account the user feedbacks in the learning phase.

In this paper, we are interested in introducing a preliminary work on the knowledge discovery module and we propose a new methodology for the exploration of unlabeled data in an unbalanced data sets. To do this we suggest a new approach to find anomalous behaviors where individuals are represented as time series with a fixed length. By anomalous[4] behavior, we mean a behavior different from the behaviors **known** as **normal**.

[3] Communications Fraud Control Association

[4] In this paper, we also use the term 'outlier' as a synonym.

2 Outlier shape detection

2.1 Time series in wholesale fraud detection

Wholesale call record details (call duration, timestamps of the call, destination,...) are grouped by calling phone numbers. Thus, each phone number is represented by multivariate time series. Depending on the number of calls made, time series can have various numbers of points. Observations are kept on a fixed duration, W, (e.g. 6 hours), meaning that all time series have the same "length" (W). The fraud detection platform must be used in a batch protocol and on past telecommunications (past calls) since the billing is not performed online[5] between the operators concerned by a given call. The aim is to find abnormal behavior time series using comparison to a time series database used before to train a model. To inform the other operators that the calls related to those time series are potential frauds, the expert is responsible to assign or not a fraudulent character label to those time series. As we are interested by the global shape of the time series, our method tries to find the abnormality using the entire time series (on the whole length W) rather than searching for outlier points/subsequences in the time series. Therefore, we propose an outlier shape detection algorithm for the whole time series rather than an outlier points detection algorithm.

2.2 Related works

Outlier detection has been extremely studied in the last decades, see for instance the Aggarwal's book [4]. More specifically, for temporal data, there are surveys like the one of Gupta and al. in [5]. Figure 3, from [5], shows a perspective on the state of the art with respect to the data type aspect.

The methodology described in this paper regards time series and thus the red square of Figure 3. As explained in Section 2.1, the wholesale setting implies to work with outlier shape detection algorithms for the whole time series. The related state of the art is therefore in the blue square of the figure: "Direct Detection Time Series Outliers" and the "Unsupervised Discriminative Approaches"[6]. The latter uses three key elements: a time series on a give representation, a clustering method and a similarity measure.

[5] Fraud disputes between operators are resolved at billing time[2].

[6] To save space, we do not give details on this part of the state of the art (see [5] for more details), but in a potential extended version paper our method will be compared with competitive approaches. The reader may also note that our method uses density estimation and therefore the state of the art on density estimation could be of interest as well.

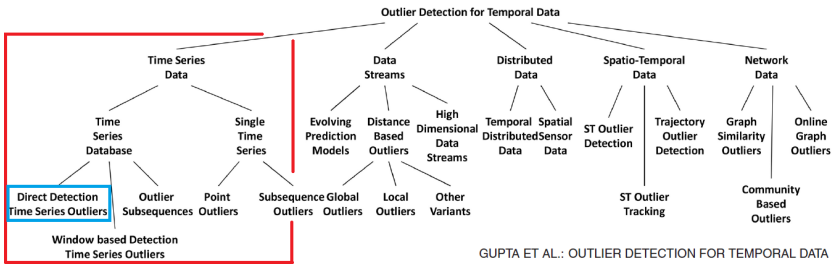


Fig. 3. A perspective on the state of the art for temporal data

3 Suggested approach of outlierness score

Our intent is to investigate a new suggestion concerning two of the three key elements of the "Direct Detection Time Series Outliers" family methods: the representation and the clustering method. We propose in a preliminary work a methodology based on the estimation of the density of normal events using a triclustering method, on a panel of time series representations and then to compare distances between this density and new individuals. The farther an individual is from the 'normal behavior' distribution, the more likely it is anomalous. The main steps of this methodology are:

1. Transform each time series in several representations;
2. Estimate the density of this set of individuals using a triclustering method;
3. Choose the best representation of time series;
4. Compute an indicator of distance to find unknown behaviors to this density.

3.1 Time series representations

In the context of classification, the study of Gay and al. [6] indicates that the choice of representation could have an important impact on the results. In our case it is impossible to know in advance which representation will be the best for density estimation vs. outliers detection. Therefore, we conduct, in Section 4, experiments using a "pool" of representations. The representation used in this first approach are the same as Gay and al. in [6]: *the original representation, the Fourier transform coefficients, the first derivative, the second derivative, the cumulated integral, the double cumulated integral and the autocorrelation function*. Each of those representations captures different information of time series. Other representations will be used in future works.

3.2 Estimation of time series density

We suggest to use a triclustering approach on the following dimensions: customers (C), timestamps (T) and a given variable (X) (for example the call duration)[7].

[7] The multivariate aspect is not studied in this preliminary work.

The input data set D is considered to be a collection of N time series, denoted S_i with $i \in 1, \ldots, N$. The key idea of triclustering with time series is that a curve can be seen as a set of points is that a curve i can be represented as a set of triplets, i.e. $S_i = \{(C_i, T_j, X_{ij}) \forall j \in 1, \ldots, m_i\}$. The total number of data points is denoted by $m = \sum_{i=1}^{N} m_i$. No assumptions are made on the time series; they could have different number of points and different timestamp values. A triclustering model aims at jointly discretizing the numerical variables T, X and at making a partition of time series by grouping the values of the categorical variable C. The objective of such models is to estimate $P(X, T, C)$, the joint probability of C, T and X. The output of the model gives groups of time series which are characterized by a bivariate discretization which estimates $P(X, T|C)$.

Amongst the state of the art of the triclustering methods we decided to use the MODL one [7]. MODL estimates the constant piecewise joint density of C, T and X with a multidimensional data grid. The optimal grid M is found by a bottom-up greedy optimization of a Bayesian criterion which makes a trade-off between accuracy and robustness of the model:

$$cost\,(M) = -log\,(p\,(M)\,p\,(D|M))$$

From the information theory point of view, this criterion can be interpreted as the encoding length of the grid plus the length of the dataset D, knowing M, according to the Minimum Description Length principle developed by Rissanen in [8]. This method is advantageous for unsupervised learning since it is free of user parameters.

3.3 Indication about the representation used

The MODL method gives a criterion named "level" (a compression rate value between 0, for a null compression gain, and 1, for a perfect compression gain) which indicates the quality of the density estimation made. One triclustering will be made per representation. Ideally, the triclustering maximizing the quality of the density estimator (value of the level) will be the best, i.e. it would also most likely provide the best separation between known normal time series and outlier time series.

3.4 Outlier score

The found partitioning gives the relationship between clusters of behaviors and groups of variables and time intervals. Thus, the outlier behavior of a customer can be linked to a specific interval of time and a set of variables. For a given new time series (not present when training the model) the purpose will be to compute how far this time series appears with respect to the model trained.

To do this we use the similarity criterion which has been defined to assess the proximity between a new time series and the clusters of a MODL coclustering model in [9]. This criterion evaluates the decrease of the level (Section 3.3), when a time series is merged within a particular cluster. This decrease can be linked to

the impact on the intra-cluster inertia of adding a new individual to its nearest cluster. Therefore, the less typical an individual from its nearest cluster, the higher the score.

4 Experimental setting and results

This section presents the data sets, the validation protocol and the results of the proposed methodology. The implementation of our method is based on the tool Khiops [10] and on R scripts for the automation of the methodology.

Our fundamental assumption is that a time series assigned to an unknown class is more likely to be anomalous than if it has been recognized as a member of a known 'normal class'. In the following experiments, we test this hypothesis by using supervised data sets where we emulate the normal/abnormal aspect of the behaviors by choosing a class as 'normal behaviors' and another as 'abnormal behaviors'. Our goal is to answer three questions :

1. Is a triclustering useful to find outlier-shape?
2. Can we find a good time series representation to detect outliers and help the triclustering?
3. Is the quality of density estimation of the triclustering correlated with the obtained results?

4.1 Protocol

As often in cyber criminality contexts, it is difficult to share data and results for confidential reason. Therefore, we experimented our method on 9 data sets from the UCR [11], offering a wide variety in terms of number and length of time series.

Each chosen dataset is composed of two classes of time series. As our model learns on a training dataset only composed of expected normal data, the train-test setting is not the one predefined in [11]. We use the following protocol to build our train-test setting:

1. Choose the majority class as normal data;
2. Choose the minority class as anomalous data;
3. Draw 70% of normal data as train data;
4. Keep remaining data as test data.

The evaluation of the score provided by the method for each representation is made with the AUC on test data. This indicator measures how well the score sort the individuals of the different classes, i.e. a good AUC that individuals from the minority class tend to have the highest scores.

4.2 Results and discussion

Performances in term of AUC (rounded to the second decimal) are reported in Table 1 for each representation (TS: original time series, DV: first derivated, ddv: second derivated, IV: cumulated integral, IIV: double cumulated integral, PWS: Fourier transform, ACF: auto-correlations). The best result for each dataset is written in bold. We detail below the results in view to the three questions questions set in Section 4.

Dataset	Train	Test	L	ROC AUC							
				TS	DV	DDV	IV	IIV	PWS	ACF	Best
Wafer	4481	2134	152	0.98	1.00	**1.00**	0.99	0.99	0.98	0.99	1.00
ECG5000	2043	1201	140	**0.99**	0.98	0.78	0.93	0.91	0.90	0.99	0.99
Computers	175	325	720	0.32	0.51	0.44	0.60	0.54	0.50	**0.64**	0.64
MoteStrain	479	793	84	**0.96**	0.95	0.84	0.85	0.66	0.61	0.82	0.96
PhalangesOutlinesCorrect	1188	567	80	0.67	0.64	0.65	0.62	0.55	**0.71**	0.63	0.71
ItalyPowerDemand	384	712	24	0.90	0.68	0.76	0.60	0.43	0.64	**0.92**	0.92
SonyAIBORobotSurface	384	377	70	0.54	0.64	0.61	**0.87**	0.79	0.58	0.66	0.87
Phalanx	235	641	80	0.40	0.30	0.31	0.41	**0.55**	0.52	0.44	0.55
WormsTwoClass	105	153	900	0.77	0.67	0.56	**0.78**	0.72	0.35	0.72	0.78

Table 1. Description of time series data sets and comparison of different reprentations. (L=Length). The column "Best gives" the best results per line for the AUC.

Is a triclustering useful to find outlier-shape? The answer seems positive: At least one AUC is better than random for each dataset (except phalanx with an AUC for double cumulated integral of 0.55).

Can we find a good time series representation to detect outliers and help the triclustering? The answer is clearly yes. The last column of the Table 1 indicates that we may find in the (small) pool of representation a representation which performs well. But no representation emerges as better that **all** others as the results of Gay and al. in [6].

Is the quality of density estimation of the triclustering correlated with the obtained results? For reasons of space limitation, we cannot give the Table of the **level** values (see Section 3.2) for all representations and data sets, but contrary to intuition, we do not find a positive correlation between the level value and the AUC. Thus the answer is negative. It appears that the method produces clusters of time series that are too specific in some cases. For example, the original representation of computers (training of 175 time series) is partitioned in 174 clusters. Despite the fact that this behavior does not necessarily imply bad results (e.g. the same problem is observed in WormsTwoClass), it is not desirable and may induce a greater variability and worst results than what could be expected.

Discussion : The experiments provide results that are consistent with the assumption that one should observe a larger score in general for classes which were not used in training. Those results are encouraging enough to pursue our efforts in this direction. In fact, by choosing the right threshold (which is a challenge by itself) for predictions one could obtain results nearly as good as in

classification (e.g. comparing our results for Wafer, SonyAIBORobotSurface and ItalyPowerDemand with Gay and al.[6], our AUC implies an accuracy, with a threshold well-chosen, nearly as good as with supervised models). Several challenges were revealed by our results. First, there is, for the moment, no way to find a good representation of the data without using a test data set which will not be available in many production applications. This is an important issue since different representations of the same dataset can lead to significant differences between AUC (e.g. ItalyPowerDemand AUC varies from 0.92 to 0.43).

5 Conclusion

This paper has presented a fraud detection platform for telecom wholesale traffic and introduced a preliminary work on an exploration module. The methodology used is based on outlier detection techniques to detect anomalous unlabeled results. The main key features of this module are: (1) use of a non-parametric method, (2) based on the ability to learn an estimator of the data density and (3) consuming a minimal amount of the expert time. The results are encouraging. They are close to those of supervised models provided by Gay et al. Works still needs to be done, notably to find a way to select/build a good representation for anomaly detection without knowing labels.

References

1. CFCA: 2017 Global Fraud Loss Survey. Survey Results, Communications Fraud Control Association (2018)
2. I3 Forum: I3f Fraud Classification. White paper 3, I3Forum (May 2014)
3. Veeramachaneni, K., Arnaldo, I., Bassias, C., Li, K., Cuesta-Infante, A.: AIˆ2: Training a Big Data Machine to Defend. In: AIˆ2: Training a Big Data Machine to Defend. (April 2016) 49–54
4. Aggarwal, C.C.: Outlier Detection. Springer (2015)
5. Gupta, M., Gao, J., Aggarwal, C.C., Han, J.: Outlier detection for temporal data: A survey. IEEE Trans. Knowl. Data Eng. **26**(9) (2014) 2250–2267
6. Gay, D., Guigourès, R., Boullé, M., Clérot, F.: Feature Extraction over Multiple Representations for Time Series Classification. In: New Frontiers in Mining Complex Patterns, Cham, Springer International Publishing (2014) 18–34
7. Guigourès, R.: Utilisation des modèles de co-clustering pour l'analyse exploratoire des données. PhD thesis, Université Paris I, Panthéon-Sorbonne (2013)
8. Rissanen, J.: Modeling by shortest data description. Automatica **14**(5) (1978) 465 – 471
9. Guigourès, R., Boullé, M., Rossi, F.: Discovering patterns in time-varying graphs: a triclustering approach. Advances in Data Analysis and Classification (Oct 2015)
10. Boullé, M.: Khiops: outil d'apprentissage supervisé automatique pour la fouille de grandes bases de données multi-tables, available at `www.khiops.com`. In: 16éme Journées Francophones Extraction et Gestion des Connaissances, EGC. (2016) 505–510
11. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The ucr time series classification archive (July 2015) `www.cs.ucr.edu/~eamonn/time_series_data/`.

Motion Segmentation by Semi-Supervised Classification in Dynamic Scenery

Petr Pulc[1], Oliver Kerul-Kmec[2], Tomáš Šabata[2], and Martin Holeňa[1]

[1] Institute of Computer Science of the Czech Academy of Sciences, Prague, Czech republic
(pulc,martin)@cs.cas.cz
[2] Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic
(keruloli,sabattom)@fit.cvut.cz

Abstract. Automatic description of multimedia content heavily relies on the ability to discover a structure in such data. As our current focus is given to efficient multimedia indexing, we are mainly interested in discovery and segmentation of foreground objects from the background and their respective description or classification.
Although many approaches based on Convolution Neural Networks have emerged lately, they are usually executed on all frames from the media separately which is, to our belief, wasteful and poorly scalable. On the other hand, methods based on visual Simultaneous Localisation and Mapping (visual SLAM) utilise the temporal structure of the motion picture to extract at first a model of the environment and objects in the scene and later pass these models to methods for object description.
In this paper, we will discuss the first two parts of the visual SLAM – motion tracking and segmentation. While many approaches impose strict restrictions in the segmentation phase to filter motion tracking outliers, we introduce restrictions to the motion tracking itself. Such approach enables us to use off-the-shelf semi-supervised classification methods in the motion segmentation phase without explicit outlier filtering.

Keywords: feature detection, motion tracking, motion segmentation

1 Introduction

With the ever-increasing amount of multimedia content, higher resolution and framerate, the requirement for faster multimedia description approaches is more significant than ever. Although the CPU power is more readily available and the image processing tools utilise GPUs much better, these advances are commonly used in favour of devising more complex processing methods rather than improving the existing approaches.

The Simultaneous Localisation and Mapping (SLAM) approach, originating from the computer vision and robotics research [15], is on the other hand aimed at rapid scene reconstruction that allows using high-level scene and object information for successive description. Also, such object description does not have

to be carried out on each frame, opposed to the approach of convolutional neural networks, such as [26]. In an ideal case, the reconstructed objects can be described once and simply tracked throughout the following frames.

To this end, SLAM proposes a pipeline consisting of a motion tracking, motion segmentation and 3D reconstruction to obtain the visual representation of individual objects from consecutive frames and map them into a 3D space. In this paper, we will focus on the first two phases of the SLAM pipeline, as 2D reconstruction is currently sufficient for our goal, whereas 3D reconstruction of static objects [13, 19, 20, 21, 22, 33] and 3D tracking of dynamic objects [2, 11, 24, 25, 34] is extensively studied.

The major challenge is that we deploy visual SLAM in dynamic environments. Therefore, in the motion segmentation step, we need to account for the fact, that not only is the camera moving through the environment, but also that the individual objects are moving independently. As a result, nor the approach based on 8-point correspondency under epipolar geometry discussed in [16] nor the 5-point relative pose resolver [23] can be used directly. They may be utilised to derive the motion of camera (ego-motion) once we know that the segmented objects are static, but we have not segmented the objects yet.

Moreover, the objects may not be rigid. In such cases, the movement of individual objects cannot be modelled by a simple homography from one frame to another. On the other hand, interest point matching based solely on the similarity of their description tends to produce many outliers that would need to be filtered.

In our approach, we propose a method that circumvents the outliers directly in the interest point matching. To this end, we still depend heavily on the similarity of interest point descriptions, but we propose a limitation on a position of the point of interest in the incoming frame. The proposed point matching is, therefore, expected to be faster as well. Consequently, this allows much higher flexibility in selection of the motion segmenter. In our approach, we picked a semi-supervised classifier with cluster regularisation [31].

In the next section, we discuss motion tracking algorithms and propose a significant improvement of our motion tracking algorithm in Section 3. Semi-supervised classifier for motion segmentation is discussed in Section 4. Finally, preliminary results of the motion segmentation based on our motion tracker are presented in Section 5.

2 Motion Tracking

As RGB video content provides no additional information than colour intensities of the individual pixels, motion tracking is estimated by tracking of small pixel patches from one frame to another. The comparison is carried out on a single channel representing luminance of the pixel (as defined in [10]) to simplify the search for a corresponding patch in the consecutive frame. If the similarity of the patches is based directly on the luminance values from the pictures, the detected apparent motion of these patches is known as optical flow.

The optical flow can be then detected on all pixels and their corresponding patches. This results in a dense optical flow [8], which may be unstable on larger areas without visible structure and is rather expensive to compute.

To increase the stability of optical flow approaches, Shi and Tomasi proposed in [30] a method of selecting only the points from the first considered frame that are supposed to be easily traceable – for example surrounded by significant luminance changes. In combination with an iterative image registration technique proposed by Lucas and Kanade [18] that utilises an image pyramid and an iterative gradient search method, correspondences of individual proposed points of interest are gathered even if the spatial distance of the new patch is greater than one pixel.

This combination of feature detector and optical flow detector is commonly referred to as a Kanade-Lucas-Tomasi feature tracker and present a viable alternative for image sequences with subtle motion and, therefore, short motion vectors as a result. The key issue with this tracker is, however, that the algorithm tries to get correspondences for all points of interest, although these points may disappear from the frame or be occluded. Such misdetections need to be filtered by executing the feature tracker backwards and checking the stability of the proposed motion vector, which slows down the whole process.

In the report by Torr and Zisserman [32], an alternative approach to feature tracking is discussed. Instead of discovering matches based on correspondences of brightness values, they advocate the use of a robust feature descriptor to capture the local structure around the point of interest in both frames. Matching the points of interest from one picture to another is then based solely on the similarity of the interest point descriptions.

Scale-Invariant Feature Transform (SIFT) [17] and Speeded-Up Robust Features (SURF) [3] are the two most widely used feature descriptors that include their own interest point detectors. However, even the improved algorithm is still computationally expensive. For time-critical applications, points of interest detected by Features from Accelerated Segment Test (FAST) [27] combined with a Binary Robust Independent Elementary Features (BRIEF) [5] or Binary Robust Invariant Scalable Keypoints (BRISK) [14] is considered more appropriate. One such combination is the ORB: Oriented FAST and Rotated BRIEF [28] which presents a decent alternative to SIFT and SURF with significantly lower computational requirements. As the GPU accelerated implementation of ORB is already available in the OpenCV library [4], it represents a perfect feature extractor for use in real-time applications and also for our research.

To find out corresponding pairs of extracted features, a distance between the feature descriptions has to be found. SIFT and SURF descriptors are real-valued and, therefore, Euclidean or Mahalanobis distance is used. Binary descriptors (BRIEF and BRISK) simplify the distance measurement to a Hamming distance that can be computed directly with `xor` and `popcnt` CPU instructions.

The most serious issue with the matching based solely on the distance of feature descriptor is that similar points of interest on unrelated positions may be mistaken with each other, which results in false correspondences. Visual SLAM in

static environments, therefore, uses robust position estimators based on sample consensus (such as Random Sample Consensus, RANSAC [9]) to reject matches not following the picture homography.

Unfortunately, this approach is not viable in dynamic environments, as the objects moving in the foreground would also violate the homography estimated from the background and all matches on the foreground objects will be thus eliminated as outliers.

3 Hierarchical Tracking of Smooth Motion

Taking into account the limitations mentioned in the previous section and the requirement to process 4K video in real-time, our approach to the extraction of points of interest and their matching is based on the following concepts:

- Speed is crucial. Therefore GPU accelerated ORB is used for interest point detection and description. Other operations can be executed on a CPU.
- Matching of the interest points needs to be based on both the point description and the estimation of new feature position in the next frame.

To this end, our current algorithm of real-time interest point matching, making an assumption of smooth object motion, consists of following steps:

1. ORB features are computed on three level scale pyramid for the new frame.
2. A fourth virtual level containing a global homography is constructed using the RANSAC from a top-most layer on the current and the new frame.
3. Matching of interest points from the current frame (filled circles in Fig. 1) to the new one (dashed circles) is carried out for the three pyramid layers from the coarsest to the finest:
   a) For each feature from the current layer (point **a**) an estimation of new position (a_n) is computed with respect to the motion of the nearest point in the layer above (A_n) and relative previous motion of the feature ($a_{n-1} - A_{n-1}$). Thus, $a_n = A_n + a_{n-1} - A_{n-1}$. If the information on previous motion is not available, $a_n = A_n$.
   b) Matching based on the Hamming distance of the feature descriptors (represented by colour of circles) is then limited only to neighbourhood of the estimated position (grey dashed circle). Considering the situation from Fig. 1, point **d** would be selected as the best match, although point **e** has smaller Hamming distance (more similar colour) and point **c** is closer to expected position.

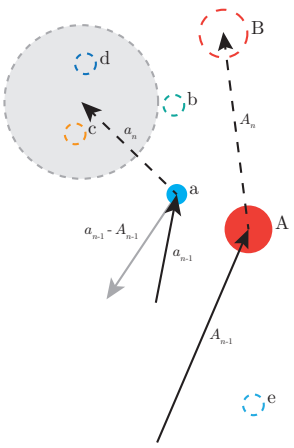


Fig. 1: Feature matching

This algorithm allows us to track motion in complex scenes with many moving objects without a need to derive the optical flow and with minimal risk of severe errors in the motion tracking.

4 Motion Segmentation based on Semi-supervised Learning

Traditionally used methods for motion segmentation in SLAM classify the gathered points of interest into two groups – static and dynamic – to base the approximation of ego-motion only on the static features. Although we need to allow more clusters in the data to segment individual objects, the basic principles of the motion segmentation are the same: to discover a mapping of individual features to the objects in considered scene and background, respectively.

Considering the state-of-the-art methods that are based on the features extracted from optical flow, only the method by Klappstein [12] can segment monocular camera recording. This method uses a graph-cut algorithm on gathered motion metric and is, therefore, computationally intensive (as outlined in [29]) but able to account for some uncertainties in the detected optical flow. Other methods (such as [1, 6, 7]) require a signal from the stereo camera to partially reconstruct depth and are, therefore, unsuitable for our purpose.

Our method takes the same underlying assumption as [12] that the individual objects can be segmented by the position and apparent motion of the points of interest. However, as the motion vectors detected by our approach already account for the significant uncertainties in the optical flow, more straightforward methods of feature space segmentation can be used. Also, our feature space has a low dimensionality (only the x and y feature coordinates, length of the motion vector and its direction), which allows us to use off-the-shelf methods directly.

When a history of motion is not available for any point of interest, unsupervised hierarchical clustering has to be used to propose segments representing individual objects in the scene. However, with the approach discussed in the previous section, such situations should not occur elsewhere than at the processing of the first two frames. During the processing of remaining frames, points of interest retain their label from frame to frame. Therefore, during the analysis of the following frame of the media, the labelled data from the previous one are used for training a semi-supervised classifier.

For motion segmentation, we propose to use the Semi-supervised Classifier with Cluster Regularisation [31] that provides a good generalisation even for small sets of labelled training data.

In our setup, we use a k-means clustering algorithm to provide an initial clustering. Based on these clusters, a pairwise penalty based on correlation is computed to disallow the boundary of final labels in high-density regions. Next, the unlabelled interest points are assigned an initial pseudo-label based on the available labels in the clusters. The computed pairwise penalty is also used to find the ten nearest neighbours of each point of interest.

For the classification, a neural network with one hidden layer is trained. The hidden layer consists of as many neurons as is the number of initial clusters, and output neurons correspond to the considered final labels. The output activation function is softmax, and the minimised loss function is cross-entropy.

5 Experimental Evaluation & Future Work

For our experimental evaluation, we prepared a set of RAW 4K videos (previews available at https://archive.org/details/motion_ds) that contain one or two objects with distinct colour on a bright background. This allowed us to extract the ground truth of pixel-to-object correspondence easily. For evaluation, the video was compressed using the H.264 codec to simulate a real-world scenario.

As our main goal is to validate the approach of motion segmentation based on our interest point matching approach and a semi-supervised classification, we devised an experiment concerning the quality of labels proposed by the classifier depending on the delay between classifier training (based on the ground truth data) and testing. To eliminate the possible dependency of the measures on the visual qualities of an individual frame, we repeated the experiment for the first five frames of the video separately.

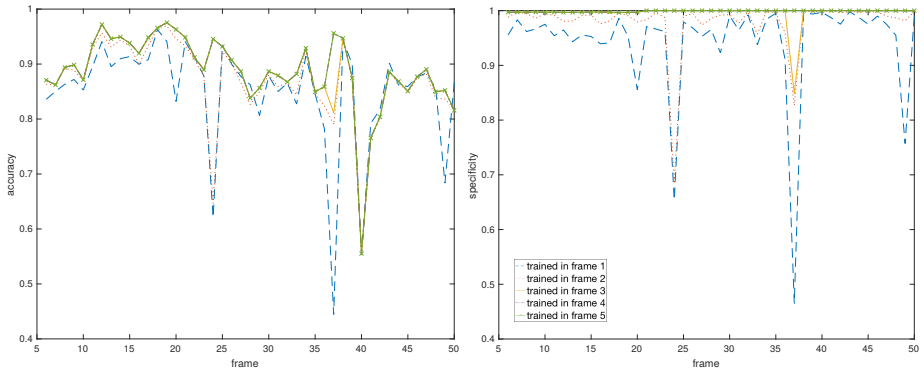


Fig. 2: Accuracy and specificity of the semi-supervised classifier with respect to the selection of training frame. File available at: https://archive.org/details/motion_ds/handheld_blurbg.mp4.

Figure 2 indicates that classifiers trained on later frames tend to have higher accuracy and specificity, but in general, the differences between classifiers trained in different frames are insignificant, which we have confirmed by the Friedman test. This behaviour is similar in all considered multimedia files and may be attributed to the fact, that the used codec introduces significant compression artefacts on the first frame (that uses only intraframe compression with limited bandwidth). Later frames have more information available and thus seem to be more stable for interest point detection.

Although motion detection based on optical flow in multimedia and motion segmentation are not novel, in this paper, we have presented a new approach in both areas that ultimately lead to the ability of real-time motion segmentation on 4K video with promising results. While the detection and tracking of 500 points on a Xeon E3-1230 CPU with the Kanade-Lucas-Tomasi algorithm takes

242 ms per frame, our approach on CPU only takes 95 ms. When nVidia 1050 Ti GPU is utilised, processing of a single 4K frame takes only 35 ms on average, which is just enough for real-time processing of 25 fps media.

In future, we intend to transfer all data-intensieve operations to a GPU and more importantly, to validate our approach on nonsynthetic datasets.

Acknowledgements

The work has been supported by the grant 18-18080S of the Czech Science Foundation (GACR).

References

[1] Pablo F Alcantarilla et al. "On combining visual SLAM and dense scene flow to increase the robustness of localization and mapping in dynamic environments". In: *ICRA*. IEEE. 2012, pp. 1290–1297.

[2] Shai Avidan and Amnon Shashua. "Trajectory triangulation: 3D reconstruction of moving points from a monocular image sequence". In: *TPAMI* 4 (2000).

[3] Herbert Bay et al. "Speeded-up robust features (SURF)". In: *Computer vision and image understanding* 110.3 (2008), pp. 346–359.

[4] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[5] Michael Calonder et al. "Brief: Binary robust independent elementary features". In: *ECCV*. Springer. 2010, pp. 778–792.

[6] Maxime Derome et al. "Moving object detection in real-time using stereo from a mobile platform". In: *Unmanned Systems* 3.04 (2015), pp. 253–266.

[7] Maxime Derome et al. "Real-time mobile object detection using stereo". In: *ICARCV*. IEEE. 2014, pp. 1021–1026.

[8] Gunnar Farnebäck. "Two-frame motion estimation based on polynomial expansion". In: *SCIA*. Springer. 2003, pp. 363–370.

[9] Martin A Fischler and Robert C Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *CACM* 24.6 (1981), pp. 381–395.

[10] ITU-R. *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios*. Recommendation. Geneva, Mar. 2011.

[11] Jeremy Yirmeyahu Kaminski and Mina Teicher. "General trajectory triangulation". In: *ECCV*. Springer. 2002, pp. 823–836.

[12] Jens Klappstein et al. "Moving object segmentation using optical flow and depth information". In: *Pacific-Rim Symposium on Image and Video Technology*. Springer. 2009, pp. 611–623.

[13] Georg Klein and David Murray. "Parallel tracking and mapping on a camera phone". In: *ISMAR*. IEEE. 2009, pp. 83–86.

[14] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. "BRISK: Binary robust invariant scalable keypoints". In: *ICCV*. IEEE. 2011.

[15] H. Lim, J. Lim, and H. J. Kim. "Real-time 6-DOF monocular visual SLAM in a large-scale environment". In: *ICRA*. 2014, pp. 1532–1539. DOI: 10.1109/ICRA.2014.6907055.

[16] H Christopher Longuet-Higgins. "A computer algorithm for reconstructing a scene from two projections". In: *Nature* 293.5828 (1981), p. 133.

[17] David G Lowe. "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2 (2004), pp. 91–110.

[18] Bruce D Lucas, Takeo Kanade, et al. "An iterative image registration technique with an application to stereo vision". In: (1981).

[19] Etienne Mouragnon et al. "Generic and real-time structure from motion". In: *British Machine Vision Conference 2007 (BMVC 2007)*. 2007.

[20] Etienne Mouragnon et al. "Generic and real-time structure from motion using local bundle adjustment". In: *Image and Vision Computing* 27.8 (2009).

[21] Etienne Mouragnon et al. "Real time localization and 3d reconstruction". In: *CVPR*. Vol. 1. IEEE. 2006, pp. 363–370.

[22] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system". In: *IEEE Transactions on Robotics* 31.5 (2015), pp. 1147–1163.

[23] David Nistér. "An efficient solution to the five-point relative pose problem". In: *TPAMI* 26.6 (2004), pp. 756–770.

[24] Kemal E Ozden, Konrad Schindler, and Luc Van Gool. "Multibody structure-from-motion in practice". In: *TPAMI* 32.6 (2010), pp. 1134–1141.

[25] Hyun Soo Park et al. "3D reconstruction of a moving point from a series of 2D projections". In: *ECCV*. Springer. 2010, pp. 158–171.

[26] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement". In: *arXiv* (2018).

[27] Edward Rosten and Tom Drummond. "Machine learning for high-speed corner detection". In: *ECCV*. Springer. 2006, pp. 430–443.

[28] Ethan Rublee et al. "ORB: An efficient alternative to SIFT or SURF". In: *ICCV*. IEEE. 2011, pp. 2564–2571.

[29] Muhamad Risqi U Saputra, Andrew Markham, and Niki Trigoni. "Visual SLAM and Structure from Motion in Dynamic Environments: A Survey". In: *ACM Computing Surveys (CSUR)* 51.2 (2018), p. 37.

[30] Jianbo Shi and Carlo Tomasi. *Good features to track*. Tech. rep. Cornell University, 1993.

[31] Rodrigo GF Soares, Huanhuan Chen, and Xin Yao. "Semisupervised classification with cluster regularization". In: *IEEE transactions on neural networks and learning systems* 23.11 (2012), pp. 1779–1792.

[32] Philip HS Torr and Andrew Zisserman. "Feature based methods for structure and motion estimation". In: *IWVA*. Springer. 1999, pp. 278–294.

[33] Changchang Wu. "Towards linear-time incremental structure from motion". In: *3DV*. IEEE. 2013, pp. 127–134.

[34] Enliang Zheng et al. "Joint object class sequencing and trajectory triangulation (jost)". In: *ECCV*. Springer. 2014, pp. 599–614.

Learning from temporal data: feature formation and prediction for categorical time series

Eman Awad and Fintan Costello

School of Computer Science,
University College Dublin,
Belfield, Dublin 6.
eman.awad@ucdconnect.ie, fintan.costello@ucd.ie

Abstract. Time-series forecasting of categorical data involves taking a temporally ordered series of categorical or nominal events and predicting the next event in the series. We describe a model of categorical time series forecasting based on frequentist probability. This model, related to cognitive models of prediction and probability estimation, learns from this type of temporal data by forming predictive features from the observed categorical series and using those features to predict the next event. We test this model in a Markov Chain Monte Carlo environment and show that its predictions are close to optimal.

1 introduction

Time series forecasting of categorical data involves taking a temporally ordered series of categorical or nominal events or labels, such as

$$A, B, S, A, B, S, A, -, S, A, -, S, -, A, B, S, A, B, A, B \tag{1}$$

and predicting the next event in the series (the next label). Since categorical data typically have no meaningful numerical ordering (and so no trend line and no average), most standard time-series forecasting approaches (e.g.the family of models based on autoregression) do not apply to categorical time-series. Forecasting models for categorical time-series are relatively rare in the literature, at least in comparison to models forecasting ordered or numerical data [1]. We describe a novel model of categorical time-series forecasting based on frequentist probability. This model builds on computational accounts of people's probability estimation [4–6] and is designed to form predictive features from the observed categorical series and uses these features to predict the next event in the series. We also describe a Markov Chain Monte Carlo environment for assessing the accuracy of categorical time-series forecasting models. We test our model in this environment and show that its predictions are close to optimal.

2 Background

Theoretically, any temporally ordered series of events can be represented by a high-order Markov chain: a chain of order n, for some value of n [14]. Markov

chains are systems that transition sequentially from one state to another according to certain probabilistic rules, producing a given event at each state. An n-order Markov chain is a model in which the probability of occurrence of a given event at time t depends on the values of the last n observed events at times $t-n$ to time $t-1$. The current state, in an n-order Markov chain, consists of the values of the last n events, which means that with m distinct categorical events a Markov model of order n consists of m^n distinct states, each with m distinct transition probabilities (one for each possible subsequent event). Such a model thus has m^{n+1} distinct free parameters (the m transition probabilities for each of m^n states) whose values must be estimated from the observed series of categorical events.

If there are long-range temporal dependencies between events in a given series, a relatively high value of the order parameter n is required to form a Markov representation of that series. This is a problem, since the number of free parameters (transition probabilities) in an n-order Markov chain is exponential with order: even with just 4 categorical events (*A,B,S,_*) as in (1), with $n = 4$ the number of free transition probability parameters that must be estimated is $4^{4+1} = 1024$. Given this, the main focus in approaches to categorical time-series representation has been been on state reduction for Markov chains [3, 14].

3 Our Model

Our model of categorical time-series forecasting takes an alternative approach: constructing features from observation of a given time series, identifying features which represent statistically reliable predictions, and then combining these 'predictive features' to give an overall predicted probability for the next event in the series. Prediction, in this model, follows a fundamental view in the computational modelling of cognitive processes, which says that many aspects of cognition are based on feature formation, feature use, and constructive feature combination. In this the model is different from the standard approach in categorical time-series prediction, which focuses on the fitting of reduced Markov models to categorical time-series data.

Our model operates by storing a series of N most recent events and forms features by combinging those events. Each such feature consists of an antecedent event A, a consequent event S, and a time interval L between them. Each feature records two counts: a count k, equal to the number of times consequent S has occurred at time L after antecedent A in the observed time series, and a count x, equal to the number of times any event at all has occurred at time L after antecedent A in the series. Each feature also holds an estimated conditional probability $P(S|A)$ (the estimated chance of seeing S at time L, given that A has just occurred), which is simply equal to k divided by x. The antecedent A in a given feature may be a single event (e.g. the label A as a predictor of the next event, in our example in (1)), or may be a combination of events occurring over time (e.g the consecutive labels *A,B* as a predictor of the next event).

The model operates by forming such features, identifying which of these features represent statistically reliable relationships between antecedent and consequent, and then combining the conditional probability of these statistically reliable predictive features to give an overall measure of the probability of a given event at the next time t.

To decide whether a given feature describes a statistically reliable relationship between antecedent A and consequent S, our model follows the hypothesis-testing approach of standard frequentist probability theory. In the case of a single event as antecedent A, we have two hypotheses: a null hypothesis (that there is no relationship between A and S; under this hypothesis the probability of seeing S after A is simply equal to the base probability of event S, $P(S)$) and an alternative hypothesis (that there is a reliable relationship between A and S; under this hypothesis the probability seeing S after A is given by $P(S|A)$). The probability of obtaining k instances of S in a sample of size x, given a base probability p, is given by

$$Bin(k, x, p) = \binom{x}{k} p^k (1-p)^{x-k} \tag{2}$$

and so if $Bin(k, x, P(S))$ is less than some significance level c, our model concludes that the null hypothesis is false and the alternative hypothesis, that there is a reliable predictive relationship between A and S, is true. The model thus marks this relationship between A and S as a statistically reliable predictor.

This model has two free parameters: c, the statistical significance level used to identify reliable predictors, and N, the number of recent events that are stored and used to form features. We can give an example of the model's operation for our example series in (1), taking $c = 0.05$ and $M = 3$. In that series there are 20 events in total, of which 5 are S (so $P(S) = 0.25$). There are 3 occurrences of B followed one step later by S, and 1 occurence of B followed one step later by a different event. To test the hypothesis that the occurence of B predicts the occurence of S at the next time step, we calculate the probability of seeing 4 occurences of B, 3 of which are followed by S, if S was occurring at its base rate probaiblity of 0.25: $Bin(3, 4, 0.25) = 0.0469$. Since this probability is less than our significance criterion of $c = 0.05$ we conclude that there is a statistically reliable relationship between B and S. Similarly, there are 5 occurrences of A followed two steps later by S, and 1 occurence of A followed two steps later by a different event. To test the hypothesis that the occurence of A predicts the occurence of S two steps later, we calculate the probability of seeing 6 occurences of A, 5 of which are followed by S, if S was occurring at its base rate probaiblity of 0.25: $Bin(5, 6, 0.25) = 0.0004$. This probability is also less than our significance criterion of 0.05 and we conclude that there is a statistically reliable relationship between A and S.

Given these results, we now have two features derived from our example series that are statistically reliable and predict the occurence of S at the next time-step (A, which predicts S after 2 steps with $P(S|A)$, and B, which predicts S after 1 step with $P(S|B)$). If we assume that these two predictive features are

independent, then according to standard frequentist probability theory the overall predicted probability is calculated by 'ORing' these independent predictions together:

$$Pr(S|A_1, \ldots, A_n) = 1 - \prod_{i=1..n} (1 - P(S|A_i)) \quad (3)$$

Applying this equation here, we get the probability of S occurring next, given the observed series in (1), of

$$Pr(S|A, B) = 1-(1-P(S|A))(1-P(S|B)) = 1-(1-0.83)(1-0.75) = 0.96 \quad (4)$$

This 'ORing' of predicted probabilities is only valid if the predictors A and B are independent: that is, if the presence or absence of the B predictor has no effect on the predictive probability $P(S|A)$, and if is the presence or absence of the A predictor has no effect on the predictive probability $P(S|B)$. If predictors A and B are not independent, then their joint presence represents a complex predictive feature $AthenB$.

3.1 Complex Features

We now consider the situation where the antecedent is itself a complex event made up of two sub-events: event A as antecedent and event B as a consequent. This complex event $AthenB$ is itself an antecedent of our consequent S. To decide whether this complex antecedent $AthenB$ reliably predicts consequent S, our model again follows the hypothesis-testing approach of standard probability theory. Here we have three possible null hypothesis / alternative hypothesis tests. In all these tests we take k to represent the number of times consequent S has occurred at time L after antecedent $AthenB$ in the observed time series, and x to represent the number of times any event at all has occurred at time L after antecedent $AthenB$ in the series.

Our first test involves the null hypothesis that there is no relationship between $AthenB$ and S; under this hypothesis the probability of seeing S after $AthenB$ is simply equal to the base probability of event S, $P(S)$. As before, if $Bin(k, x, P(S)) < c$, say, our model concludes that this null hypothesis is false, and the alternative hypothesis, that there is some reliable relationship between $AthenB$ and S is true.

Our second test considers the form of this relationship between $AthenB$ and S, by asking whether the occurrence of A alone predicts the occurrence of the consequent S. Our null hypothesis here that there is no *additional* relationship between $AthenB$ and S, beyond that between A alone and S; under this null hypothesis the probability of seeing S after $AthenB$ is simply equal to the probability of S given A, $P(S|A)$. If $Bin(k, x, P(S|A)) < c$, our model concludes that this null hypothesis is false and the alternative hypothesis, that there is some reliable relationship between $AthenB$ and S beyond the relationship between A and S, is true.

Finally, our third test asks, in just the same way, whether the occurrence of B alone predicts the occurrence of the consequent S . Our null hypothesis that

there is no *additional* relationship between $AthenB$ and S, beyond that provided by B. If $Bin(k, x, P(S|B)) < c$, our model concludes that this null hypothesis is false and the alternative hypothesis, that there is some reliable relationship between $AthenB$ and S beyond the relationship between B and S, is true.

Each of these three tests involves asking whether the observed pattern of occurence of antecedent $AthenB$ followed by consequent S can be explained by the base probability rates $P(S), P(S|A)$ and $P(S|B)$. If this pattern of occurrence cannot be explained by any of these three probabilities, we deduce that the presence of the complex feature $AthenB$ leads to reliably different predictions for the consequent S than those that would be obtained by considering the two components A and B as independent predictors. This means that the feature $AthenB$ is itself a distinct, statistically reliable predictor of the occurrence of S: given that $AthenB$ has occurred, the probability of S is given by $P(S|AthenB)$ (rather than by $P(S), P(S|A)$ and $P(S|B)$). By contrast, if there is no additional relationship between the feature $AthenB$ and S beyond that given by A and B, we can conclude that A and B are independent predictors of S and the normatively correct predictive probability for S, given these independent predictors, is obtained by ORing their probabilities as in (4).

4 Testing the Model

Recall that any process producing a series of categorical events can be represented by an nth order Markov chain (for some value of n). Any such chain thus represents a realistic generative model of categorical time-series production. We use this view of Markov chains as generative models to assess our model in a Markov Chain Monte Carlo environment [9], as follows.

First we construct a generative n-th Markov chain with $m = 4$ (4 distinct categorical events) and $n = 4$ (the current state consists of the last 4 categorical events). There are $4^4 = 256$ distinct states in this chain, each with 4 transition probabilities. For each state these 4 transition probabilities are assigned random values, normalised so their sum for that state equals 1. These randomly assigned transition probabilities are fixed when the Markov chain is constructed, and remain unchained throughout the generative process.

Next we use this Markov chain to generate a large series of categorical events. We first pick 4 initial events at random, representing the initial state of our Markov Chain. We identify the 4 transition probabilities associated with that state, and choose one transition (one new event) at random, proportional to its transition probability from the current state in the Markov Chain. The selected event is added to our series. The new state of our Markov model now consists of the 4 most recent events (three previous events and the event that was just added to the series), and the cycle repeats.

These events are fed, one at a time, to our categorical forecasting model. We run the model here with parameters $c = 0.05$ and $N = 4$. The forecasting model runs through the series, forming statistically reliable simple and complex

Table 1: For a series of probability 'buckets' (ranges **R**; column 1) this table shows the number of times our forecasting model predicted that an event would occur with a probability that fell into range **R** (column 2), and of those predictions, the number of times the predicted event actually occurred (column 3). Column 4 gives the ratio of these two numbers, and column 5 gives the difference between this ratio and the centerpoint of the range **R**.

Range **R**	Model makes prediction in **R**	Predicted event occurs	Ratio	Difference
0.05 - 0.10	1393	193	0.14	0.06
0.10 - 0.15	2268	401	0.18	0.05
0.15 - 0.20	2600	516	0.19	0.02
0.20 - 0.25	3016	6463	0.21	0.01
0.25 - 0.30	3901	1094	0.28	0.005
0.30 - 0.35	3341	1005	0.3	0.02
0.35 - 0.40	1806	597	0.33	0.04
0.40 - 0.45	788	272	0.35	0.08
0.45 - 0.50	307	117	0.38	0.09
0.50 - 0.55	171	64	0.37	0.15
0.55 - 0.60	60	32	0.53	0.04
0.60 - 0.65	25	13	0.52	0.10

predictive features and making predictions for the next event to occur at each time step.

After an initial training phase (the first $10,000$ events in our Markov series) we continue running the forecasting model and the Markov generator for an additional $50,000$ event test series. We gather, at each time step, both the model's predicted probability of next occurrence for each of the m available categorical events, and the transition probability in our Markov chain from the current state to each of those m categorical events. We assess the model in terms of the accuracy of its probability estimates for the next event, by comparing forecasting model predictive probabilities against Markov chain trainsition probabilities. If the model is accurate, these should agree. We also assess the model by gathering together all cases where the model predicts that an event will occur with probability in some range R. If the model is accurate, the proportion of those predicted events that did actually occur should be in or near the probability range R.

4.1 Results

There was a reliable correlation between, across the $50,000$ test items, between the forecasting model's predictive probabilities and Markov chain transition probabilities ($r > 0.99, p < 0.0001$), indicating that forecasting model was accurately representing markov transitions. To test the model's predictions directly (rather than comparing against probabilities in the generative Markov model),

We grouped the forecasting model's predictive probabilities into a series of 'buckets' or ranges $\mathbf{R}$ (so one bucket would hold all cases where the model predicted some event with a probability between 0.05 and 0.1, another would hold all cases where the model predicted some event with a probability between 0.1 and 0.15, and so on). For each bucket we counted the number of times the predicted event actually occurred. If the model is predicting events correctly we would expect that, if we have C cases where the model predicts that some event will occur with a probability between 0.05 and 0.1, then the predicted event should actually occur in somewhere around $0.05 \times C$ or $0.1 \times C$ of those cases.

Table 1 shows the results of this analysis of model prediction. As this table shows, there was a reliable correlation between the range in which the model predicted an event will occur and the actual rate or ratio of occurrence of that event ($r = 0.97, p < 0.0001$). Table 1 also shows that model accuracy is related to sample size (ranges where the model made many predictions have smaller predictive differences than those where the model made fewer predictions). Finally, Table 1 also shows a noticable pattern of 'regression': events tend to occur at probabilities close to, but higher than, to those predicted by the model. This pattern of regression is a central characteristic of human probabilistic reasoning.

4.2 Conclusions

We have described a simple model of categorical time-series prediction based on the formation of statistically reliable predictive features. This model is interesting because it derives from a standard view in computational models of cognition, which is that the formation and use of features is a fundamental aspect of human probabilistic reasoning and cognition. This model represents a computational implementation of a cognitive account of probabilistic reasoning proposed by Costello & Watts [4] which has explained many puzzling aspects of people's probabilistic judgement and inference.

The core proposal in the Costello & Watts account is that people's probabilistic judgement follows frequentist probability theory, but is subject to random variation or noise. Costello & Watts conducted several experiments examining people's probability estimates for a range of different events; the results provided evidence that a wide range of systematic biases observed in people's probability judgement could be explained and predicted by patterns of regression caused by this random variation [4–8]. The simple model we have described here implements this frequentist approach in detail. Notably, the model of probabilistic prediction described here also shows just this pattern of regression (as seen in Table 1) and so is consistent with all of these results on systematic bias in people's probabilistic reasoning. In future work, we plan to test this model using data from selected psychological experiments in the areas of categorisation and classical conditioning [2] [10] [11] [12] [13] [15].

We hope that presentation of this model will lead to useful cross-fertilisation between two currently separate areas of research: cognitive models of human learning and predictive processing, and computational analysis of temporal data.

References

1. Angers, J.F., Biswas, A., Maiti, R.: Bayesian forecasting for time series of categorical data. Journal of Forecasting **36**(3), 217–229 (2017)
2. Balsam, P.D., Gallistel, C.R.: Temporal maps and informativeness in associative learning. Trends in neurosciences **32**(2), 73–78 (2009)
3. Berchtold, A., Raftery, A.E.: The mixture transition distribution model for high-order markov chains and non-gaussian time series. Statistical Science pp. 328–356 (2002)
4. Costello, F., Watts, P.: Surprisingly rational: probability theory plus noise explains biases in judgment. Psychological review **121**(3), 463 (2014)
5. Costello, F., Watts, P.: Peoples conditional probability judgments follow probability theory (plus noise). Cognitive Psychology **89**, 106–133 (2016)
6. Costello, F., Watts, P.: Invariants in probabilistic reasoning. Cognitive psychology **100**, 1–16 (2018)
7. Costello, F., Watts, P.: Probability theory plus noise: Descriptive estimation and inferential judgment. Topics in cognitive science **10**(1), 192–208 (2018)
8. Costello, F., Watts, P., Fisher, C.: Surprising rationality in probability judgment: Assessing two competing models. Cognition **170**, 280–297 (2018)
9. Geyer, C.J.: Practical markov chain monte carlo. Statistical science pp. 473–483 (1992)
10. Kamin, L.: Predictability, surprise, attention, and conditioning. in ba campbell & rm church (eds.), punishment and aversive behavior (pp. 279-296). New York: Appleton-Century-Crofts (1969)
11. Love, B.C.: Comparing supervised and unsupervised category learning. Psychonomic bulletin & review **9**(4), 829–835 (2002)
12. Pizzo, M.J., Crystal, J.D.: Time-place learning in the eight-arm radial maze. Animal Learning & Behavior **32**(2), 240–255 (2004)
13. Posner, M.I., Keele, S.W.: Retention of abstract ideas. Journal of Experimental psychology **83**(2p1), 304 (1970)
14. Raftery, A.E.: A model for high-order markov chains. Journal of the Royal Statistical Society. Series B (Methodological) pp. 528–539 (1985)
15. Smith, D.J., Minda, J.P.: Thirty categorization results in search of a model. Journal of Experimental Psychology: Learning, Memory, and Cognition **26**(1), 3 (2000)

An Intelligent Approach for Managing Uncertainty in Temporal Databases: First Steps

Manel Chehibi[1], Aymen Gammoudi[2,3], and Allel Hadjali[3]

[1] University of Manouba, Tunisia
[2] LARODEC/ISG, University of Tunis, Tunisia
[3] LIAS/ENSMA, Poitiers, France

Abstract. Temporal databases often incorporate vague/imprecise, subjective, and uncertain temporal information. This uncertainty and vagueness of temporal data poses a problem regarding the representation of temporal knowledge and temporal reasoning. In this paper, we present an approach based on the theory of belief functions for the representation and management of imprecision and uncertainty of temporal information.

Keywords: Temporal Databases · Uncertainty · Imprecision · Theory of belief functions.

1 Introduction

Temporal informations are often expressed in an imprecise, subjective and uncertain way, for example, "well after early 20" or "to late 30". Imprecision means that temporal data are not specific instants but fuzzy time intervals. As to the uncertainty, it describe a partial knowledge about the truth.

In the literature, many works have been proposed to represent, manage, and reason about imperfect temporal information. In [1], Nagypál and Motik proposed a model based on fuzzy sets that extend Allen's work to fuzzy time intervals. However, their approach does not consider imprecise temporal relations. In [2], Schockaert and De Cock proposed a model based on the concept of fuzzy orders. This model allows to model imprecise and gradual interval relations. Hadj Salem et al. [3] proposed an approach based on belief function theory for the representation and management of uncertainty of temporal relations. In the framework of this theory, modeling the belief of an agent about the uncertain relationship $r_{a,b}$ between two instants a and b is expressed by means of a mass of belief $m_{a,b}$. The major disadvantage of this approach is that it considers only the uncertain temporal relations between specific instants, whereas vague and uncertain temporal information are often represented by fuzzy time intervals. Gammoudi et al. [4] proposed an approach based on fuzzy set theory for the representation and management of imprecision of the temporal data. In this paper, the qualitative temporal formalism used for temporal information processing is Allen's algebra. The major problematic point of this approach is that it only considers imprecise time intervals, so it does not allow to deal with

uncertain time intervals, while temporal information are often subjective and uncertain. In [4], Allel et al. proposed a set of operations that make it possible to reason about uncertain temporal relations. These operations are : inversion, composition, combination and negation operation.

In this paper, we propose an approach based on the theory of belief functions for the modeling and management of imperfect temporal data in the Allen's interval algebra context. In fact, the belief functions theory provides strong tools for modeling and combining uncertain, imprecise and subjective information and it is specialized to model the uncertainty more than probability theory: unlike the theory of probability, Dempster-Shafer theory is able to make the distinguish between the case of total ignorance and the case of equiprobable hypotheses.

This paper is organized as follows: Section 2 recalls some basic concepts of the theory of belief functions and Allen's interval algebra, Section 3 details the proposed approach to represent and manage imperfect temporal information before concluding in section 4.

2 Background

In this section, we give a brief recall on the theory of belief functions (This section is mainly taken from my article [10]), and on Allen's relations between temporal intervals.

2.1 Theory of belief functions

The theory of belief functions, also called Dempster-Shafer theory, was first introduced by Dempster [6] and mathematically formalized by Shafer [7]. This theory models imprecise, uncertain and missing data.

In the theory of belief functions, a *frame of discernment*, noted $\Theta = \{H_1, ..., H_N\}$, is a set of N exhaustive and mutually exclusive hypotheses $H_i, 1 \leq i \leq N$. only one of them is likely to be true.

The *power set*, $2^\Theta = \{A/A \subseteq \Theta\} = \{\emptyset, H_1, ..., H_N, H_1 \cup H_2, ..., \Theta\}$, enumerates 2^N sub-assemblies of Θ. It includes not only hypotheses of Θ, but also, disjunctions of these hypotheses.

The true hypothesis in Θ is unknown; thus, a degree of belief is assessed to subsets of 2^Θ reflecting our degree of faith on the truth of each subset of 2^Θ.

A *basic belief assignment (bba)*, also called *mass function*, is noted m^Θ and defined such that:

$$\begin{aligned} & m^\Theta : 2^\Theta \rightarrow [0,1] \\ & m^\Theta(\emptyset) = 0 \\ & \sum_{A \subseteq \Theta} m(A) = 1 \end{aligned} \tag{1}$$

The mass $m^\Theta(A)$ represents the degree of belief on the truth of $A \in 2^\Theta$. When $m^\Theta(A) > 0$, A is called *focal element*.

The belief function, noted bel^{Θ}, measures the amount of faith on the proposal A. It is the sum of masses of subsets implying A. This function is defined as :

$$\begin{aligned} &bel^{\Theta} : 2^{\Theta} \rightarrow [0,1] \\ &bel^{\Theta}(A) = \sum_{B \subseteq A, B \neq \emptyset} m^{\Theta}(B) \end{aligned} \tag{2}$$

The plausibility function, noted pl^{Θ}, measures the maximum amount of faith with which we found A plausible. It is the sum of masses of the proposals that do not contradict A (whose intersection with A is not the empty set). This function is defined by:

$$\begin{aligned} &pl^{\Theta} : 2^{\Theta} \rightarrow [0,1] \\ &pl^{\Theta}(A) = \sum_{A \cap B \neq \emptyset, B \subseteq \Theta} m^{\Theta}(B) \end{aligned} \tag{3}$$

In the theory of belief functions, combination rules are proposed to merge distinct mass functions in order to produce a more reliable information. It consists on building an unique mass function by combining several elementary mass functions arising from multiple distinct sources of information.

Dempster's rule of combination [1] is the first rule that merges several mass functions provided by distinct and independent sources. The combination of two mass functions $m^{\Theta}_{S_1}$ and $m^{\Theta}_{S_2}$ provided by S_1 and S_2 is given as follows:

$$m^{\Theta}_{1 \oplus 2}(A) = (m^{\Theta}_1 \oplus m^{\Theta}_2)(A) = \begin{cases} \dfrac{\sum_{B \cap C = A} m^{\Theta}_1(B) \times m^{\Theta}_2(C)}{1 - \sum_{B \cap C = \emptyset} m^{\Theta}_1(B) \times m^{\Theta}_2(C)} & \forall A \subseteq \Theta,\ A \neq \emptyset \\ 0 & \text{if } A = \emptyset \end{cases} \tag{4}$$

The reliability of an evidential information is not always insured. In fact, an evidential data can be supplied by a partially reliable or an unreliable source. In order to take the source's reliability into account, its beliefs are discounted proportionally to its reliability. Let $\alpha \in [0,1]$ be the reliability of a source S_1 and m^{Θ} a mass function provided by S_1. The *discounting* of m^{Θ} produces ${}^{\alpha}m^{\Theta}$ defined by:

$$\begin{cases} {}^{\alpha}m^{\Theta}(A) = \alpha \times m^{\Theta}(A) & \text{if }, \forall A \subset \Theta \\ {}^{\alpha}m^{\Theta}(\Theta) = 1 - \alpha \times (1 - m^{\Theta}(\Theta)) \end{cases} \tag{5}$$

In the theory of belief functions, decision is generally made using *pignistic probabilities* [8]. The pignistic probability, noted $BetP^{\Theta}$, is deduced from m^{Θ} as follows:

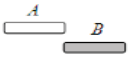
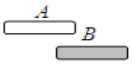
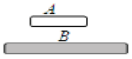
$$BetP(H_i) = \sum_{\substack{A \in 2^{\Theta} \\ H_i \subset A}} \frac{1}{\mid A \mid} m^{\Theta}(A) \qquad \forall H_i \in \Theta \tag{6}$$

where $\mid A \mid$ is the number of hypotheses which train it.

2.2 Allen's interval algebra

Allen's interval algebra [9] is one of the most known and used formalisms in temporal reasoning. A significant part of the work on temporal representation and reasoning is concerned with time intervals. It is an algebra based on 13 primitive and mutually exclusive relations that can be applied between two time intervals $A = [a, a^{'}]$ and $B = [b, b^{'}]$. These relationships are: before (b), after (bi), meets (m), met by (mi), overlaps (o), overlapped by (oi), starts (s), started by (si), during (d), contains (di), finishes (f), finished by (fi) and equals (e). Their meaning are expressed in table 1. Each of these relations corresponds to a particular order of the four bounds of the two intervals. For example, the statement A overlaps B (A o B) corresponds to $(a < b) \wedge (b < a^{'}) \wedge (a^{'} < b^{'})$.

Table 1. Allen's temporal relations

Relation	Inverse	Signification	Relations between bounds
A b B	B bi A		b > a'
A m B	B mi A		a' = b
A o B	B oi A		b > a ∧ a' > b ∧ b' > a'
A d B	B di A		a > b ∧ b' > a'
A s B	B si A		a = b ∧ b' > a'
A f B	B fi A		a > b ∧ b' = a'
A e B	B e A		a = b ∧ a' = b'

3 Proposed Approach

The uncertainty and vagueness of temporal data poses a problem regarding the representation of temporal knowledge, temporal reasoning and also in the context of query answering systems to deal with temporal questions which are about uncertain and imprecise temporal data. For example, as shown in the database example in [5], the statement that the discovery of cones rocks took place in 2003

is vague, imprecise, uncertain and it leads to many possible hypotheses: i.e., the discovery of cones rocks may have occurred in early 2003, or at the end of 2003 or during 2003 etc. Therefore, in this section, we present an approach based on Dempster-Shafer theory for modeling the uncertainty, inherent in temporal data more precisely inherent in relations between temporal intervals. Then we propose a three basic rules that enables us to reasoning about temporal uncertain relations.

3.1 Uncertainty modeling

Let R the frame of discernment represents the set of possible relations that can be hold between two interval I_1 et I_2.

$$R = \{b, bi, m, mi, o, oi, s, si, d, di, f, fi, e\}$$

We refer to an element of this set as $r \in R$. The belief of an agent on subsets of the frame of discernment R is represented by a basic belief assignment (BBA) m^R. This BBA m^R is defined by the following mapping:

$$\begin{array}{c} 2^R \rightarrow [0,1] \\ A \rightarrow m^R(A) \end{array}$$

and satisfied:

$$\begin{array}{c} m^R(\emptyset) = 0 \\ \sum\limits_{A \subseteq R} m(A) = 1 \end{array}$$

The mass m(A) represents the degree of belief assigned to the element $A \in 2^R$. For example, let I_1 the time interval that represents the discovery of cone rocks. Let I_2 the time interval that represents the year of discovery which is 2003, $I_2 = [01/01/2003 - 31/12/2003]$. An archaeologist can express that mass allocated to possible relations between I_1 and I_2 are: $m_{I_1,I_2}(o) = 0,2$; $m_{I_1,I_2}(s) = 0,6$ $m_{I_1,I_2}(\{s,d\}) = 0,1$; and $m_{I_1,I_2}(\{R\}) = 0,1$.

The state of knowledge of an agent on the uncertain relation can also be represented by two vector:

- A credibility vector of the form: $Bel_{I_1,I_2} = (Bel^{r_1}_{I_1,I_2}, Bel^{r_2}_{I_1,I_2}, ..., Bel^{r_n}_{I_1,I_2})$ with $Bel^{r_i}_{I_1,I_2} > 0, r_i \in R$, $i \leq |R|$, measure the belief (2) in the relation r_i.
- A plausibility vector of the form: $Pl_{I_1,I_2} = (Pl^{r_1}_{I_1,I_2}, Pl^{r_2}_{I_1,I_2}, ..., Pl^{r_n}_{I_1,I_2})$ with $pl^{r_i}_{I_1,I_2} > 0$, measure the plausibility (3) of the relation r_i.

Let's go back to our previous example, the state of knowledge of the archaeologist can then be represented by the vector $Bel_{I_1,I_2} = (Bel_{I_1,I_2}(o) = 0.2,\ Bel_{I_1,I_2}(s) = 0.6)$ and the vector $Pl_{I_1,I_2} = (Pl_{I_1,I_2}(o) = 0.3,\ Pl_{I_1,I_2}(s) = 0.8, Pl_{I_1,I_2}(d) = 0.2)$.

3.2 Reasoning about temporal relation

In this section, we propose three basic operations which are inversion, composition and combination. These operation enables us to deduce new temporal information.

Operation of inverse

The inversion operation ($\sim$) is an operation allows us to deduce the two vector Bel_{I_2,I_1} and Pl_{I_2,I_1}, from the two known vector Bel_{I_1,I_2} and Pl_{I_1,I_2}. This operation exchange the relation r_i with its inverse $\widetilde{r}_i$ (temporal relations and their inverses are shown in section 2.2).
– $Bel_{I_2,I_1} = (Bel^{\widetilde{r}_1}_{I_1,I_2}, Bel^{\widetilde{r}_2}_{I_1,I_2}, ..., Bel^{\widetilde{r}_n}_{I_1,I_2})$
– $Pl_{I_2,I_1} = (Pl^{\widetilde{r}_1}_{I_1,I_2}, Pl^{\widetilde{r}_2}_{I_1,I_2}, ..., Pl^{\widetilde{r}_n}_{I_1,I_2})$

For instance, considering the two vectors of the previous example it's easy to deduce that:
– $Bel_{I_1,I_2} = (Bel_{I_1,I_2}(oi) = 0.2,\ Bel_{I_1,I_2}(si) = 0.6)$
– $Pl_{I_1,I_2} = (Pl_{I_1,I_2}(oi) = 0.3,\ Pl_{I_1,I_2}(si) = 0.8, Pl_{I_1,I_2}(di) = 0.2)$.

Operation of combination

The purpose of this operation is to combine distinct mass functions in order to produce a more reliable information. Let $m^R_{s_1}$ and $m^R_{s_2}$ two mass functions arising from two distinct sources or experts s_1 and s_2 and representing respectively their beliefs on the nature of relation between two time intervals I_1 and I_2. The combination operation ($\oplus$) allows combining the two elementary mass functions $m^R_{s_1}$ and $m^R_{s_2}$ provided by s_1 and s_2 into a single mass function m^R. The Dempster combination rule, presented in Section 2.1, is then applied.

$$m^R = m^R_{s_1} \oplus m^R_{s_2}$$

For example, two archaeologists gave their opinions on the temporal relation between two time intervals I_1 and I_2. According to the first archaeologist s_1: $m^{s_1}_{I_1,I_2}(\{s,d\}) = 0,7$; $m^{s_1}_{I_1,I_2}(d) = 0,1$; and $m^{s_1}_{I_1,I_2}(\{R\}) = 0,2$. According to the second archaeologist s_2: $m^{s_2}_{I_1,I_2}(\{d,f\}) = 0,8$; $m^{s_2}_{I_1,I_2}(d) = 0,1$ and $m^{s_2}_{I_1,I_2}(\{R\}) = 0,1$. In order to get at the end one more reliable mass function, the two mass functions provided by the two archaeologists s_1 and s_2 are combined (see table 1). The resulting combined BBA is: $m^R_{I_1,I_2}(d) = 0.75$, $m^R_{I_1,I_2}(\{d,s\}) = 0.07$, $m^R_{I_1,I_2}(\{d,f\}) = 0.16$ and $m^R_{I_1,I_2}(\{R\}) = 0.02$.

It should be noted that it is important to take the sources' reliability into account using the discounting rule (5) before performing the combination operation. In the previous example, we assumed that the sources of information are entirely reliable.

Table 2. Illustrative example of the combination operation

m^{s_1}/m^{s_2}	$m^{s_2}(\{d,f\})=0,8$	$m^{s_2}(d)=0,1$	$m^{s_2}(\{R\})=0,1$
$m^{s_1}(\{s,d\})=0,7$	$m(d)=0.56$	$m(d)=0.07$	$m(\{s,d\})=0.07$
$m^{s_1}(d)=0,1$	$m(d)=0.08$	$m(d)=0.01$	$m(d)=0.01$
$m^{s_1}(\{R\})=0,2$	$m(\{d,f\})=0.16$	$m(d)=0.02$	$m(R)=0.02$

Note that also we can apply the pignistic transformation (6) that compute the pignistic probabilities $BetP$ from the resulting combined BBA, $m^R_{I_1,I_2}$, in order to decide about the relation between I_1 and I_2.

Let's go back to our previous example, after computing the pignistic probability $BetP$, we obtain: $BetP(d)=0,87$; $BetP(s)=0.04$ and $BetP(f)=0.09$. We then deduce that $I_1\ d\ I_2$ which means that the discovery of cone rocks took place during the time interval $I_2=[01/01/2003-31/12/2003]$.

Operation of composition

Given the bba $m^R_{I_1,I_2}$ which expresses nature of the uncertain relation between two time intervals I_1 and I_2, and the bba $m^R_{I_2,I_3}$ which expresses nature of the uncertain relation between two time intervals I_2 and I_3, the operation of composition $\bigotimes$ allows us to estimate, using the Dempster's rule of combination (4), the uncertain temporal relation r_{I_1,I_3} that may hold between the two time intervals I_1 and I_3.

For example, Let's consider the following two bba's modeling the uncertain knowledge about the relations between I_1 and I_2 and between I_2 and I_3:
$m_{I_1,I_2}(b)=0,7$; $m_{I_1,I_2}(\{b,o\})=0,1$; $m_{I_1,I_2}(\{R\})=0,2$
$m_{I_2,I_3}(b)=0,6$; $m_{I_2,I_3}(\{b,m\})=0,3$; $m_{I_2,I_3}(\{R\})=0,1$
The combination of these two BBA is shown in table 2.

Table 3. The combination of BBAs

$m_{I_1,I_2}/m_{I_2,I_3}$	$m_{I_2,I_3}(b)=0,6$	$m_{I_2,I_3}(\{b,m\})=0,3$	$m_{I_2,I_3}(\{R\})=0,1$
$m_{I_1,I_2}(b)=0,7$	$m(b)=0.42$	$m(b)=0.21$	$m(b)=0.07$
$m_{I_1,I_2}(\{b,o\})=0,1$	$m(b)=0.06$	$m(b)=0.03$	$m(\{b,o\})=0.01$
$m_{I_1,I_2}(\{R\})=0,2$	$m(b)=0.12$	$m(\{b,m\})=0.06$	$m(\{R\})=0.02$

Thus, the mass of the derived relation is: $m^R_{I_1,I_3}(b)=0,91$; $m^R_{I_1,I_3}(\{b,o\})=0,01$; $m^R_{I_1,I_3}(\{b,m\})=0,06$ and $m^R_{I_1,I_3}(\{R\})=0,02$.

4 Conclusion

In this article, we propose an approach based on the theory of belief functions for modeling and managing imprecise, subjective, and uncertain temporal information. The qualitative temporal formalism used for temporal information processing is Allen's algebra. In this work, we model the knowledge of agent about the nature of temporal relation between temporal intervals using a mass function. From this mass we can conclude the most plausible relationship that can be hold between two interval of time. We proposed also a set of operations that allow us to infer new temporal information and deduce new temporal relation. The work described in this paper is part of the theoretical foundation for building an intelligent query answering system to temporal databases. As future work, we will propose an approach for handling gradual and imprecise database queries (which have fuzzy and flexible time criteria).

References

1. Nagypal, G., Motik, B.: A fuzzy model for representing uncertain, subjective, and vague temporal knowledge in ontologies. In Proc. of OTM Confederated International Conferences On The Move to Meaningful Internet Systems, 2888, 906–923. (2003)
2. Schockaert, S., Cock, M. D.: Temporal reasoning about fuzzy intervals. Artificial Intelligence, 172, 1158-1193. (2008)
3. El Hadj Salem, N., Hadjali, A., Gammoudi, A. and Ben Yaghlane, B.: An evidential approach for managing temporal relations uncertainty. In Nguyen, N. et al. (Eds), Proceedings of the 8th International Conference on Computational Collective Intelligence Technologies and Applications (ICCCI'16), Vol. 9875, Springer, Halkidiki, pp. 98–107.
4. Gammoudi, A., Hadjali, A., Ben Yaghlane, B.: Fuzz-TIME: an intelligent system for managing fuzzy temporal information. Int. J. Intell. Comput. Cybern. 10(2), 200–222 (2017) Cybern. 10 (2) (2017)
5. . Hadjali, A., Dubois, D., Prade, H.: A possibility theory-based approach to the handling of uncertain relations between temporal points. In: Proceedings of the 11th International Symposium on Temporal Representation and Reasoning, TIME 2004, pp. 36–43. IEEE (2004)
6. A. P. Dempster: Upper and lower probabilities induced by a multiple valued mapping. The Annals of Mathematical Statistics, 1967.
7. G. Shafer: A mathematical theory of evidence. Princeton University Press, 1976.
8. P. Smets. Decision making in the tbm: the necessity of the pignistic transformation. International Journal of Approximate Reasonning, 2005.
9. James F. Allen. Maintaining knowledge about temporal intervals. Communications of the ACM, 26(11):832–843, 1983.
10. Chehibi, M., Chebbah, M., Martin, A.: Independence of Sources in Social Networks. Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Foundations - 17th International Conference, IPMU 2018, Cádiz, Spain, June 11-15, 2018, Proceedings, PartI, pp. 418–428. (2018)