Evaluating Subjective Accuracy in Time Series Pattern-Matching Using Human-Annotated Rankings

Philipp Eichmann
peichmann@cs.brown.edu

Emanuel Zgraggen
ez@cs.brown.edu

Brown University
Department of Computer Science
Providence RI, United States

ABSTRACT

Finding patterns is a common task in time series analysis which has gained a lot of attention across many fields. A multitude of similarity measures have been introduced to perform pattern searches. The accuracy of such measures is often evaluated objectively using a one nearest neighbor classification (1NN) on labeled time series or through clustering. Prior work often disregards the subjective similarity of time series which can be pivotal in systems where a user specified pattern is used as input and a similarity-based ranking is expected as output (query-by-example). In this paper, we describe how a human-annotated ranking based on real-world queries and datasets can be created using simple crowdsourcing tasks and use this ranking as ground-truth to evaluate the perceived accuracy of existing time series similarity measures. Furthermore, we show how different sampling strategies and time series representations of pen-drawn queries effect the precision of these similarity measures and provide a publicly available dataset which can be used to optimize existing and future similarity search algorithms.

Author Keywords

Time series; crowd sourcing; result ranking; sketching; information retrieval; sampling; pen & touch.

ACM Classification Keywords

H.3.3. Information Storage and Retrieval: Query formulation; H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

INTRODUCTION

Data in the form of time series can be found in any domain. Stock prices, medical data, trajectories of objects, crime statistics, "Quantified Self" data, temperatures or sales figures can all be represented as data streams over time and even higher dimensional data are straightforward to visualize using two-dimensional line graphs. Such visualizations enable humans to interpret and compare time series data. However, visually identifying patterns within large time series or across multiple time series is a tedious and error prone task. Since traditional query languages (e.g. SQL) are often not well suited for this type of temporal pattern search, researchers have published an abundance of papers introducing pattern-matching algorithms.

Modern hardware, which supports pen and/or touch input allows users to swiftly sketch query patterns for time series data. Consider a medical researcher who wants to find patients with a specific disease who have a decreasing platelet count or a stock-trader who wants to find stocks following a certain trend. While such queries are cumbersome to specify through standard UI controls, drawing an upward or downward line on a tablet through a pen-stroke or a touch-gesture is simple and intuitive. These hand-drawn patterns, however, are not exact and therefore, the retrieved results should be displayed as similarity-based rankings, where the first item is the time series most similar to the input sketch.

Figure 1 shows a screenshot of TimeSketch, a pen and touch-based application that allows users to query time series through such hand-drawn patterns and displays results as similarity-based rankings. We implemented this prototype to test and experiment with different time series pattern-matching algorithms and sampling strategies. Through an informal user evaluation of TimeSketch, we realized that algorithms oftentimes produce perceivably inaccurate results, as depicted in the screenshot.

The accuracy of many existing pattern-matching algorithms have been evaluated through nearest neighbor classification (1NN) with much less attention given to how humans gauge the results of such similarity searches. This does not come as a surprise; label prediction with no further distinction within a label group can be sufficient to compare the overall performance of different techniques. However, in cases where the expected output of a search is a similarity-based ranking, a more detailed comparison is required. These rankings also need to align with human intuition and a user's perceived similarity. "Good" matches, as defined by a distance metric of an algorithm, might not coincide with the subjective similarity. This is especially important for the top k matches, whereas dissecting time series which are dissimilar to the query pattern is more difficult and arguably less important to a user. Due to the lack of datasets that contain hand-drawn patterns and human-annotated similarity rankings based on those patterns, it is currently hard to understand the type of patterns

IUI'15, March 29 - April 01 2015, Atlanta, GA, USA.

ACM 978-1-4503-3306-1/15/03 $15.00
http://dx.doi.org/10.1145/2678025.2701379

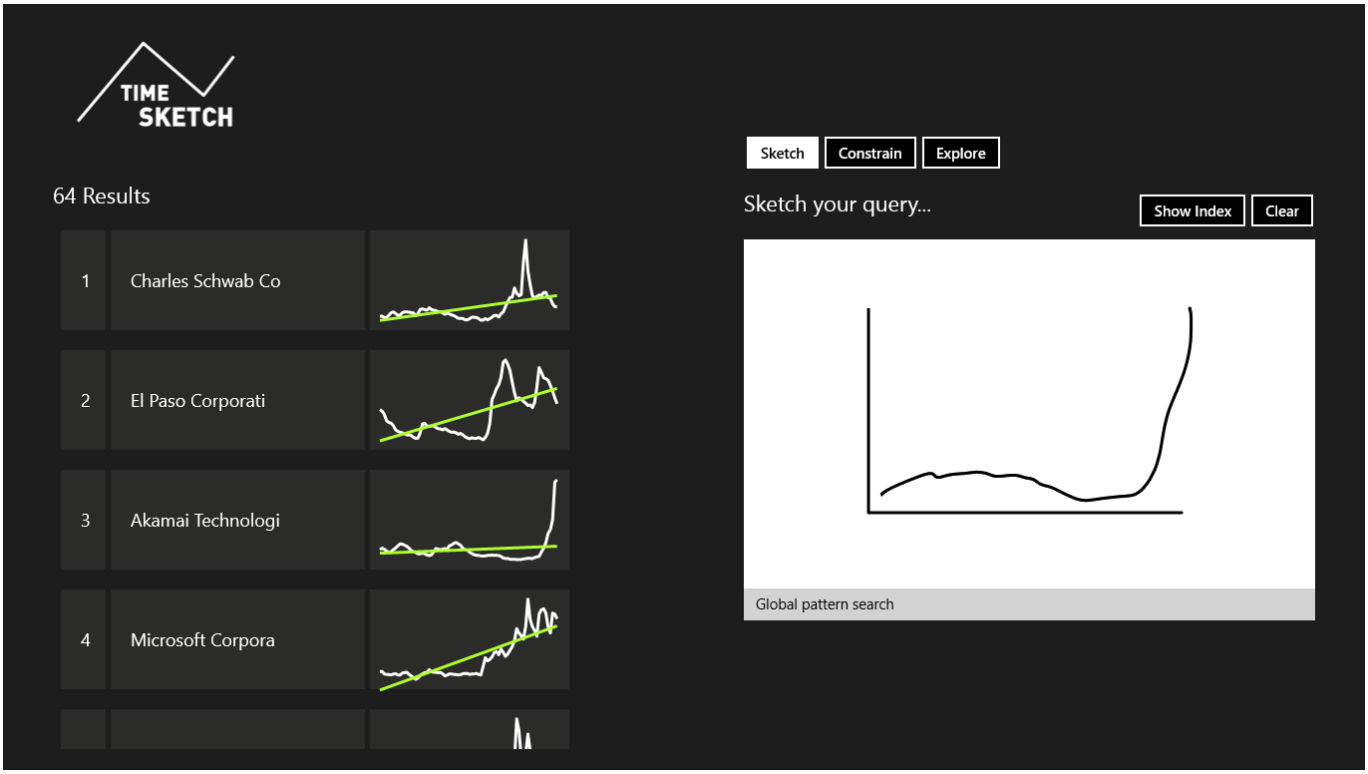

Figure 1. Screenshot of TimeSketch, a pen and touch-based application which allows users to query time series through pen or touch-drawn query patterns. Different combinations of sampling strategies and similarity measures (uniform linear downsampling to 80 points and SpADe similarity measure, in this case) sometimes return a perceivably incorrect ranking of the search results. Result number 3, for instance, should be clearly ranked first.

users draw as well as to measure the subjective accuracy of existing algorithms for such use-cases.

In this paper, we present a methodology to create human-annotated rankings of time series based on the perceived distance to a hand-drawn reference pattern. We then use these rankings to evaluate the accuracy of existing time series similarity measures, across different sampling strategies and with varying time series representations. Furthermore, we provide some analysis of our collected hand-drawn patterns and our evaluation that provide insights for possible future research efforts.

PRELIMINARIES AND RELATED WORK

In this section, we give an overview of research efforts in the areas of query-by-example for time series data, time series representations, and similarity measures. In addition, we introduce definitions that we will refer to throughout this paper. We refer the reader to [6] and [20] for a more comprehensive summary of time series representations and similarity measures.

Definitions and Notations

In the upcoming sections will use similar definitions as in [21] and [6]:

Time Series: A time series T is an ordered list $T = t_1, t_2, ..., t_n$ where each t_i is a data point in a $d-$dimensional space.

Subsequence: A subsequence $T_{i,k}$ of time series T where i is the starting index in T and $k = |T_{i,k}|$ is the length of the subsequence in number of data points.

Query Pattern: A Query Pattern Q is a subsequence used as input for similarity search whose data points correspond to either a subsequence of an existing time series or to one defined by other means (e.g. mouse/pen/touch-sketch).

Time Series Representations

An unmodified time series is called a *raw representation* [6] and its length depends on both, the time-span of the recorded data and the sampling rate. Typically, before a similarity search is executed, the two time series are converted to a different representation. This has the advantage of reducing (or sometimes also increasing) the number of data-points i.e. the dimensionality for reasons of space and search efficiency, noise reduction or simply scaling both sequences to an equal length. The GEMINI framework [8], for instance, proposes to find candidate matches by downsampling the time series to a dimension that works well with index structures. Once all candidate matches are found, their full-resolution time series is read from disk and compared with the query sequence using a distance metric such as the Euclidean Distance. This greatly speeds up the pattern-matching process as opposed to having to sequentially scan all time series.

Time series representations can be categorized as follows: data-adaptive, non-adaptive, model-based and data-dictated

[22]. An overview and description of many representations can be found in [6, 17]. In previous publications, time series representations have been solely compared to each other in terms of indexing performance using a metric called *Tightness of lower the bound (TLB)*. This is the ratio of the estimated distance between two sequences under that representation, over the true distance between the same two sequences [13]. Interestingly, it has been found that there is very little difference between different representations in terms of indexing speed [19, 6]. Moreover, it was claimed that a representation should not be chosen based on its approximation fidelity, but rather on other features such as the visual appearance. In this paper, we exploit this fact to choose a suitable segmentation technique in order to project a query pattern defined by a pen stroke to a time series using different representations. Specifically, we use variants of Perceptually Important Points (PIP) [9] and Piecewise Linear Segmentation (PLS) [14] among two other approaches which we describe in the Method section.

Similarity Measures

Similarity measures are used to compute a distance between two time series. [6] distinguishes between *lock-step*, *elastic*, *threshold-based* and *pattern-based* measures.

The simplest similarity measures are lock-step measures, which compute the distance between two time series in a one-to-one fashion, where each data point in one time series is only compared to its direct counterpart in the other. Typically, the *Euclidean Distance* is used because it penalizes larger distances more, but other Lp-norms such as the Manhattan Distance fall into this category. One of the biggest shortcomings of lock-step measures is that they are sensitive to local time shifting [4], such that even two time series that are perceived to be similar could be separated by a large distance. This makes the Euclidean Distance unsuitable for our application, as hand-drawn queries should be invariant to local time shifting and scaling. Nonetheless, we use this measure in our experiments in order to draw a comparison to other methods.

Elastic measures support shifts and scaling in time-dimension. Most notably Dynamic Time Warping (DTW), which was recently claimed to be the best measure in terms of speed and accuracy [21]. It tries to find the optimal alignment between two time series by allowing for local warping within a certain window. DTW uses the Euclidean Distance as distance measure between two points. Thus, the Euclidean Distance is essentially a special case of DTW, where the warping window size is equal to 1. Other edit distance based measures include Longest Common SubSequence (LCSS) [1, 24], Edit Sequence on Real Sequence (EDR) [23], Swale [18], Edit Distance with Real Penalty (ERP) [3]. Elastic measures are especially important for human-drawn queries as the query patterns are never exact. Due to its popularity and recent research efforts, we use DTW in our experiments.

SpADe (Spatial Assembly Distance) is a pattern-based measure that promises to handle time/amplitude- shifting and scaling and noise [4]. It works by computing features over short subsequences of two time series, called *local patterns*, finding similar patterns called *local pattern matches* in both time series and eventually calculating the shortest distance between all found matches from the start of the time series to the end. As we consider time and amplitude invariance important characteristics of similarity measures for human-drawn queries, we also employ SpADe in our experiments.

Pattern-Matching Invariance

Some similarity measures reduce variance in pattern-matching by making the measure flexible to time-shifting/scaling and amplitude-shifting/scaling, which is important in applications where the query pattern is defined by hand. Based on [2], we give a brief overview on how these and other types of invariance can be implemented.

Even though many subsequences in a dataset may look similar to a query pattern, they may be defined at different amplitude ranges or they are located at different offsets. *Amplitude invariance* can be achieved by normalizing the time series which are compared to each other. It is known that a *z-score normalization* follows the original shape of the data-points more closely than a *min/max normalization* [17].
Elastic similarity measures handle *local scaling* in time-dimension. To fix *global scaling*, on the other hand, we have to either test all possible window sizes or define a time range for a query pattern to eliminate the problem entirely. To tackle *phase invariance*, we have to test all possible alignments in time. Moreover, [2] suggests that *occlusion invariance*, the case where we want to ignore one or multiple subsequences in a pattern, can be achieve by simply ignoring the distance on those time intervals.

Defining Query Patterns

The idea of specifying time series queries in a query-by-example fashion has been around for over a decade. Typically, the goal of a similarity search is to perform a k-NN-search i.e. to return the k nearest neighbors given an input [25]. Similar to our prototype TimeSketch, QuerySketch lets users define arbitrary queries by sketching onto an empty graph, where the time scale can be adjusted by using a zoom function [26]. The Euclidean Distance between the query pattern and the time series in the dataset is used as similarity measure and each result is labeled with the distance to the query.

TimeSearcher uses the concept of timeboxes, rectangular regions that can be drawn over a time series, and uses the containing subsequences as query patterns [11]. Since timeboxes define value-ranges rather than fix-valued data points, the similarity of the specified query and the time series in a dataset is determined by an orthogonal range tree algorithm [5]. Counters are used as degree of similarity by measuring how many data points lie in the specified amplitude and time range. Even though the bulk of this paper focuses on query patterns rather then range-queries, TimeSearcher provides inspiration for further research (see Discussion and Future Work).

[10] presents TimeSearcher: Shape Search Edition (SSE), an extension of TimeSearcher where query patterns can be defined by shapes such as spikes, sinks, rises, drops, lines, plateaus, valleys and gaps. The authors describe attributes of

these shapes by which they can be identified, compared and ranked. The similarity measures we used in our evaluation are purely mathematical. However, as we note in the Discussion and Future Work section, a combination of high and low level similarity measures is a promising research direction.

Methods to evaluate time series similarity

Keogh et al. propose two ways to measure the true similarity of two time series [13], subjective evaluation and objective evaluation. The latter can be carried out by using a 1NN classification to predict a label for a time series from its nearest neighbor in a training dataset, where the classifier's distance measure is the similarity measure in question. A major drawback of this approach when evaluating similarity is that both, the training and test data must be labeled. Even though there are publicly available labeled datasets, new and unlabeled datasets cannot be tested. Furthermore, this method only measures how often the prediction of a label failed or succeeded but fails to provide a distance of how well it actually matched a query pattern, which is inevitable to create rankings. Also, it is often unclear who created the labels and based on which decisions.

Subjective evaluation relies on human judgment. A trivial way to evaluate the perceived accuracy of time series is to simply visualize matching results of different similarity measures and let people assess the quality of the algorithm. Another commonly used approach is to create a dendrogram of multiple time series using the single linkage method and let humans determine, if similar time series are reasonably linked. The authors of [15] and [14] note that the "correct" distance measure depends on user preference and the data domain. They describe an approach to adjust the similarity metric by using feedback from users. More specifically, they propose the time series to be represented as piecewise linear segments (PLS) and a weight vector to describe the relative importance of each segment. The weights are dynamically updated based on the user's relevance feedback, which allows for a custom weighting scheme for each user and domain.

METHOD

While existing time series datasets, such as [16], are suitable to benchmark classification and clustering accuracy, there are currently no datasets that provide a distance-based ranking or other means to fine-tune a similarity search algorithm which returns an ordered list of results, based on a given input sequence. The subjective accuracy of a similarity search algorithms depends on the data-domain and user preference [15, 14]. Therefore, when creating a system similar to TimeSketch, it is important to evaluate the perceived accuracy on a case-by-case basis. In this section, we show how human-annotated rankings can be created for any type of time series data based on [12] and using simple crowdsourcing tasks. Using the help of the crowd is especially advantagous in the absence of expert annotators or when annontating large amounts of time series data becomes unfeasible.

In Computer Vision, for instance, the human-perceived accuracy of image classification systems can be determined by testing the classifier against a human-annotated set of images (ground-truth). Borrowing this idea, we describe how a ground-truth for selected subsequences in a given dataset can be created using a crowdsourcing platform and use this ground-truth to compute a similarity-based ranking over all subsequences.

Creating Human-Annotated Rankings

In order to create a human-annotated ranking for a specific query, we first generate all possible subsequences in a dataset that match the length of a query pattern. We then create a set of all unordered pairs of subsequences and use a pairwise comparison between each of the subsequences to build a global ranking, according to their similarity to the query pattern.

More formally, for a given query pattern q and a dataset d, we generate a set S_q comprising all subsequences in d of the same length as q, where S_q is the set of all subsequences that will be part of the similarity ranking. Let U_{Sq} be the set of all unordered pairs in S_q. For each pair of subsequences $\langle a, b\rangle$ in U_{Sq}, we create a task on a crowdsourcing platform, where the workers' task is to decide, which of the two subsequences resembles the query pattern more (binary classification). We let the same task be answered by n distinct workers, in our case $n = 9$. The degree of agreement among all workers i.e. the probability of one subsequence being more similar to the query pattern than the other, can then be used to compute a global score of similarity. To obtain such a score for any subsequence $a \in S_q$, we can sum up the probabilities of a being more similar to q then all other subsequences in S_q.

$$score(a) = \sum_{b \in S_q} p(\langle a, b\rangle) \tag{1}$$

Where p in Equation 1 is a function that returns the probability of a being more similar to q than to the other subsequence b of a pair, according to the number of received votes in the binary classification task. To create the ranking, S_q can be sorted by the score of its elements.

Comparing Rankings

Computing a ranking from the distances provided by a similarity measure is as simple as ordering all subsequences based on their distance in ascending order. Once all rankings for all similarity measures and query pattern representations have been established for a particular query, they can be compared to the human-annotated ranking to see how well both rankings align. Given a human-annotated ranking R_H and any similarity-measure based ranking R_S, we compute the Spearman's rank correlation coefficient

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \tag{2}$$

where d_i is the difference of the ith ranking in R_S and R_H respectively and n is the total number of ranks. A Spearman coefficient of 1 or -1 indicates that the two ranking correlation perfectly (positively or negatively) which means the produced ranking is identical to the human created one. We use this correlation coefficient as a measure of how well two

ID	Dataset	Length
1	Daily reported crimes in Chicago	4,748
2	Microsoft Stock Values	5,036
3	NBA Player Scores	930
4	City Temperatures	2,920

Table 1. List of all datasets used in survey

Dataset	Question
1	Crime rate which fluctuates periodically for a couple of months
1	A period of a year, where the crime rate increased steadily
2	Find the 2008 stock market crash
2	A situation where the stock value dropped and remained low for a while
3	Find a player who started off really well at the beginning of his career, but had decreasing scores over his entire career
3	Find a period where a player consistently scored more than 20 points per game
4	Find a place in the United States where it is warmer in Winter than in Summer
4	Find places with hot months in 2013

Table 2. Example questions users tried to answer with a pen/touch-based interface.

rankings align because the distance between two ranks are squared. Thus small differences in ranks (i.e., transpositions) are less penalized than larger ones.

EVALUATION AND ANALYSIS

Using the method described in the previous section, we created 8 different human-annotated rankings through Amazon Mechanical Turk, using 8 pen-drawn query patterns over 4 different datasets and compared the accuracy of 3 different similarity measures in conjunction with 4 distinct sampling strategies.

Datasets

To avoid bias in queries towards a certain domain, we created 4 different datasets with varying lengths and context, as shown in Table 1. Dataset 1 contains daily reported crime incidents in the city of Chicago from 2001 to 2013. In the second dataset, we listed the Microsoft stock closing values of all trading days over a period of 10 years (1994-2013). Dataset 3 contains the number of points former NBA player Michael Jordan scored per game for all season games he had played. Finally, dataset 4 comprises maximum daily temperatures of 8 different cities in the United States.

Collecting Query Patterns

Instead of merely using a subsequence of a time series as query pattern, we asked 6 users to draw queries using a pen/touch-based application, which records all strokes and pen-related meta-data. Before recording a query pattern, we instructed the participants on how to use the tool and which dataset their queries should target. For instance, a user was requested to draw an arbitrary number of queries for a dataset comprising the maximum daily temperature of 8 different

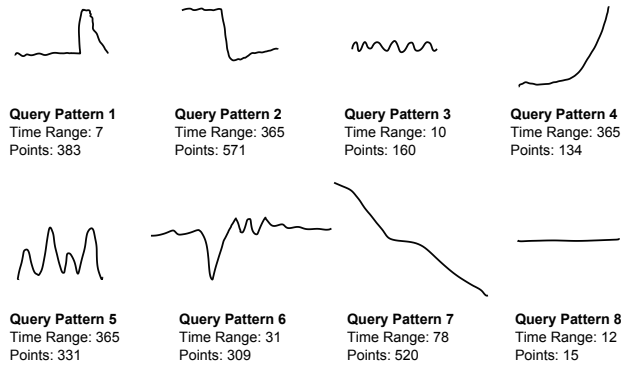

Figure 2. Eight hand-picked query patterns used to create human-annotated rankings.

cities in the United States. The user was then asked to define a pattern length in terms of number of data points (e.g. 30 days) and optionally instructed the application to draw guide lines, which some users found helpful to define their queries more precisely. Also, the participants were able to choose whether or not they wish to ignore the given minimum and maximum amplitude in their query and to change the size of the drawing area.

We recorded a total number of 112 pen-drawn queries (shown in Figure 8 at the end of this paper) over the 4 different datasets. The participants of our survey were all graduates students with no prior experience in time series analysis. For each pattern, we recorded what question the user wants to answer with her/his query, some of which can be found in Table 2.

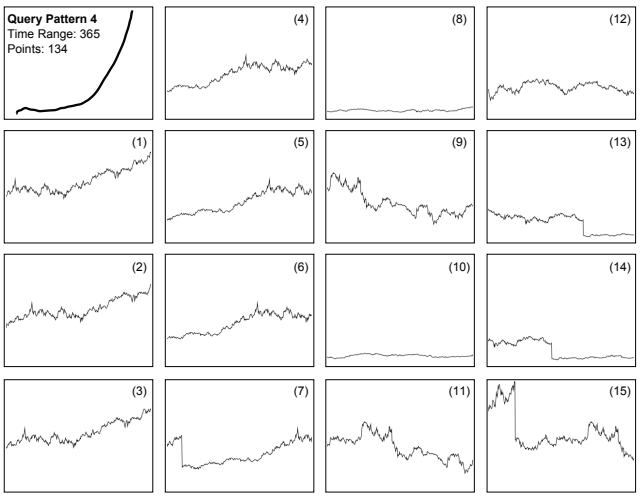

Figure 3. Example of a human-annotated ranking created through crowd-sourcing tasks.

Creating a Human-Annotated Ranking

After collecting sketches, we proceeded to creating human-annotated rankings based on our method outlined earlier in the paper. Since the number of required pairwise comparisons becomes very large for a large number of subsequences, we followed the following procedure: from the set of all recorded

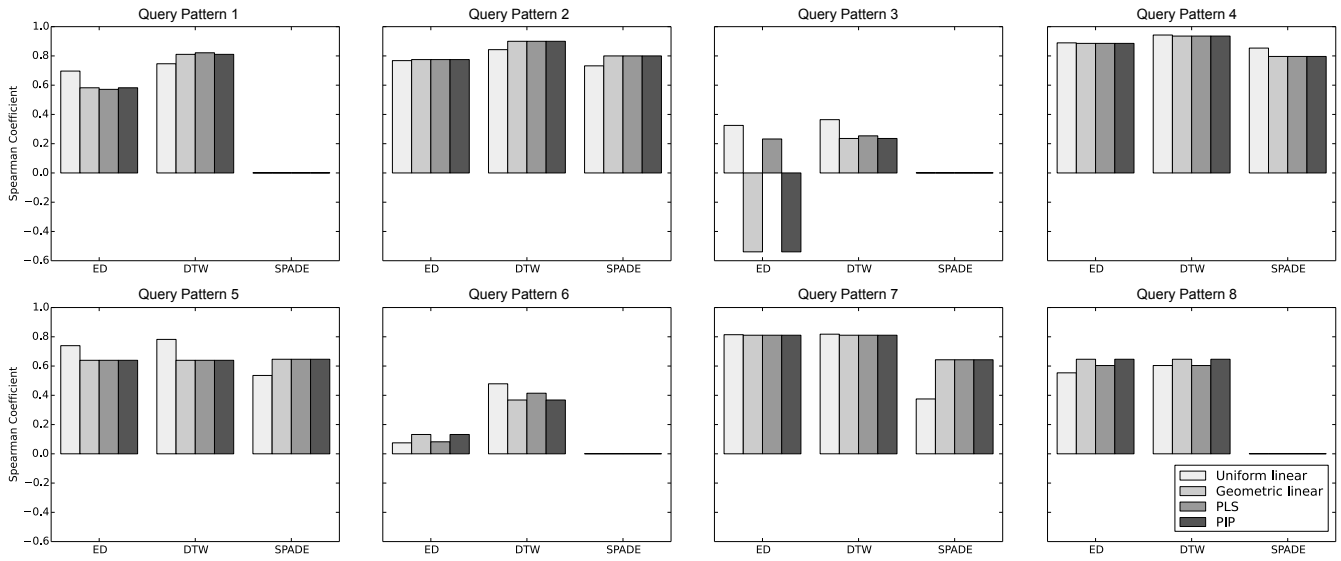

Figure 4. Spearman correlation coefficients of different algorith ms and sampling methods when compared to human-annotated rankings (1 indicating perfect positive correlation, -1 perfect negative correlation).

query patterns, we hand-picked eight (two for each dataset) to assure coverage of the most distinct types of user-drawn queries. Using a window of the size of the pattern length, we then generated all possible subsequences with step size 1. Next, we assigned each retrieved subsequence to one of 25 clusters using k-means clustering and randomly picked 4 from each cluster. This left us with 100 subsequences to be compared with the query pattern. Since this would still require us to do 44,550 pairwise comparisons ($9 \cdot \binom{100}{2}$), we used a two-task crowdsourcing strategy. In a first task, we displayed the query pattern and a plot of one of the 100 subsequences. The workers' task was to judge the similarity of two graphs by selecting a value from 1 to 10. We let each comparison between the query pattern and every subsequence be answered 5 times and then selected the 15 subsequences that resemble the query pattern most, according to the workers' median answer. Using these steps, we were able to reduce the number of required pairwise comparisons for each query pattern to 105. In a second crowdsourcing task, we employed our method of creating a human-annotated ranking. We created a set of all unordered pairs of the 15 subsequences and built a binary classification task where the query pattern is displayed along with the two subsequences of a pair. The workers' task was then to select the one subsequence of a pair which resembled the query pattern more. We let each task be answered 9 times and then computed the score of each subsequence using Equation 1.

Evaluating Similarity Search Accuracy

In order to create a similarity measure based ranking for the query pattern, we first need to transform the sketch into an evenly spaced time series. This is a delicate task, as the strokes provided by pen/touch hardware consist of a varying number of non-uniformly distributed 2D-points, whereas the time series in our datasets are evenly spaced. To conduct this transformation, we first order all points by their time stamp. We then remove all points that are not monotonically spaced in time. Next, we sample the points to match the length of the query pattern with one of the following algorithms:

Uniform Linear: A uniform linear sampling stores the values of all points in an array and performs up/down-sampling using a linear smoothing kernel. It is naive in the sense that it distorts the shape of the originally drawn pattern by directly converting an unevenly spaced time series to an evenly spaced one.

Geometric Linear: A geometric linear sampling maps each point of the resulting time series to a location between two 2D-points of the input stroke. The value at that location is computed using linear interpolation.

PLS: PLS sampling performs a piecewise linear segmentation on the input stroke as described in [14]. The value is then retrieved in the same manner as in the Geometric Linear approach.

PIP: PIP sampling applies the PIP-algorithm on the input stroke as described in [9]. As in PLS, the value is then retrieved in the same manner as in the Geometric Linear approach.

Once the query pattern is converted to a time series, we can z-normalize both the pattern and subsequences and use a similarity measure to compute the distance between them. We employ three different algorithms in our experiments: Euclidean Distance, DTW and SpADe. To avoid implementation bias, we use the original implementation from [21, 4] respectively. The results of our comparisons can be seen in Figure 4. DTW produces rankings that are closest to the human-annotated rankings. For short query patterns (1, 3,

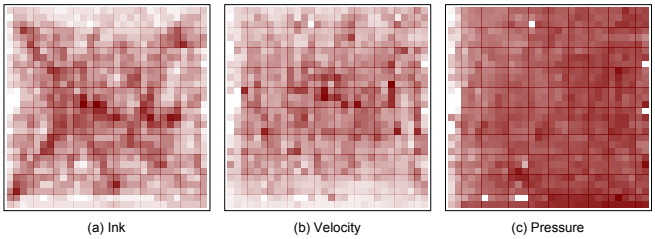

Figure 5. Heat-maps visualizing different properties across all query patterns over the normalized sketching area. Dark-red indicates a higher value.

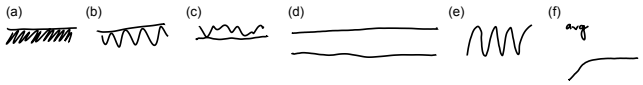

Figure 6. More involved multi-stroke query patterns that are not supported by tested time series pattern-matching algorithms.

6 in Figure 3), SpADe was unable to output meaningful distances. We verified this fact with the author, who confirmed that SpADe does not work well for very short time series. For larger time series it yields competitive results (query patterns 2, 4 and 5). The Euclidean Distance expectedly performed worse than DTW or equally well in some cases. Interestingly, it performed very badly in combination with Geometric Linear and PIP sampling in some cases (query pattern 3 and 6). We found that, especially for short time range queries, Geometric Linear and PIP sampling significantly distort the original sketch, which can lead to a comparitively large Euclidean Distance. DTW, however, manages to remedy a poorly sampled sketch to a certain extend. The naive uniform linear sampling did surprisingly well. One plausible reason for its good performance is that even though there are changes in velocity across the drawing area, as can be seen in Figure 5, the smoothing kernel makes up for the non-uniformly distributed points.

Overall, different sampling strategies do not seem to impact the accuracy of algorithms drastically. In fact, Geometric Linear, PLS and PIP actually often yield roughly the same results due to their identical interpolation method and only differ in the way the points are chosen during down-sampling.

Query Patterns

In order to get a better understanding of the types of questions users try to answer with the patterns they draw, we analyzed various properties of the sketches. Figure 5 shows some results of that analysis. (a) depicts the average ink distribution of all queries over a normalized sketching area (note that users were able to adjust the size of the sketching area). Users tend to focus on patterns that either target some extrema (corners) or align their patterns within the center region. We also recorded pen-speed as well as pen-pressure for each pattern. On average, velocity is highest within the center of the sketching area (b). We explain this through the ease-in/ease-out style that users tend to draw their patterns with. (c) suggests that there is no clear trend for pen-pressure as it appears to be uniform across the drawing area.

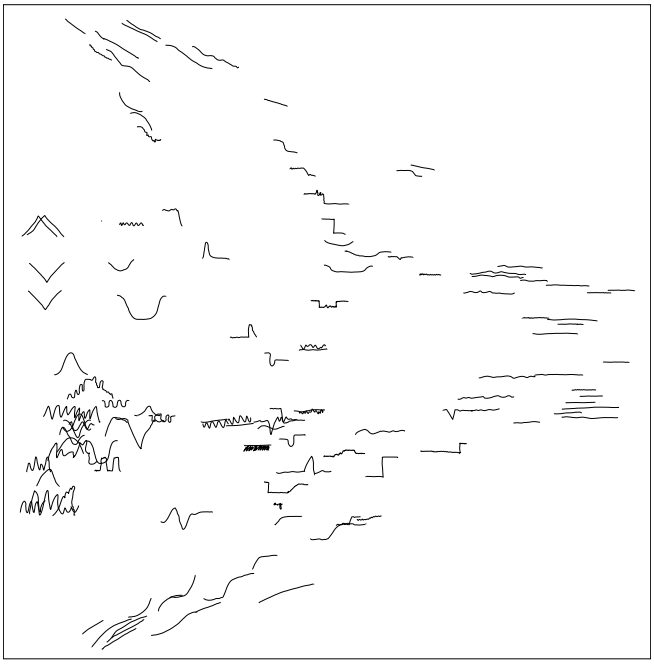

Figure 7. Automatic layout of all query sketches generated by applying PCA to the feature histograms of individual sketches.

While most of our 112 pen-drawn query patterns were simple single stroke patterns that were intended to find subsequences which followed a specific trend, some of our users wanted to search for more complex relations in the data. Questions like "Find a period where a player consistently scored more than 20 points per game" were usually drawn with multi-stroke sketches by our users. Figure 6 shows some queries with these characteristics. The sketches in (a), (b) and (c) are instances where users wanted to search for inequalities (e.g., "find everything above / below X"), while (d) depicts a range query (e.g., "find everything within X and Y"). Query (e) is an example where a user wanted to find time series which shows periodic fluctuations over a specific time period ("find all inconsistent basketball players") . However, our user did not intend to find time series that were close in shape to her sketch. (f) is a case, where a user annotated the sketch to provide some more information to the system. It is important to note that current state of the art time series pattern-matching algorithms, such as the ones we tested, do not support such queries directly. In a future effort, we aim to collect more hand-drawn sketches and try to extract a common sketching language that could be used to express other relations such as inequalities, ranges, fuzziness or fluctuations.

DISCUSSION AND FUTURE WORK

To get an intuition of the quality of the produced human-annotated rankings we inspected them visually. Although we agreed on most of the decisions made by the crowd, there were cases where we felt that a pair of subsequences should have been swapped. It is important to note that apart from a qualification test, the workers' answers were not validated or filtered using sentinel tasks or any other noise reduction technique. Also, with only 9 votes per pairwise compari-

son, chances are relatively high that some subsequences receive the same score. Increasing the number of required votes could help reducing ambiguities in generated rankings. Alternatively, multiple 9-round tasks could be averaged.

The analysis of the sampling methods used in our evaluation suggests that there are more suitable sampling methods than the ones used in our experiments. Sampling between two points artificially distorts the resulting time series. This is especially the case for short-length query pattern with high complexity. An adaptive sampling method, where the perceivably important points are aligned with the resulting time series could potentially lead to better results.

To further understand the different types of queries that users draw, we wanted to see if there are certain categories of patterns that occur across users and across datasets. Just by visually inspecting Figure 8, we can identify that for example straight-line or upward/downward trend patterns are drawn frequently. We obtain automatic clusters of such sketch categories by applying a similar approach as discussed in [7]. Each sketch is rendered as an image which we then represent by using a normalized feature histogram that is computed through a bag-of-words model over densely sampled local features. Figure 7 shows an automatic layout generated by applying dimensionality reduction (PCA) over these feature histograms. This initial experiment hints at the possibility of creating an accurate classifier to assign category labels to input sketches. Such labels could in turn be used to detect different pattern types, such as inequalities, range queries, etc. or to optimize algorithms through fast-rejection techniques.

We are planning to conduct a more rigorous study on what type of time series queries users are interested in. The results of the study could be used to define a visual vocabulary i.e. sketching-language which supports common types of queries. Because our approach of evaluating the subjective accuracy in time series pattern-matching can be extended to an entire conceptual query space, algorithms supporting these kind of queries could then be evaluated using the method proposed in this paper. Other possible areas for future research include finding better sampling/alignment methods for hand-drawn queries (possibly using weighted strokes) and combining high and low-level similarity measures to improve the perceived accuracy of pattern-searches. For instance, DTW could be used to retrieve the top n matches while shape attributes described in [10] could be used to generate a ranking over the n matches.

Our comparison of algorithms and sampling strategies through our 8 collected human-annotated rankings offers anecdotal evidence that human-rankings can differ drastically from rankings generated by algorithms. For a better understanding of the different approaches, rigorous statistical analysis is needed. Such an analysis should cover a wide range of time series from different domains and queries drawn by experts in these domains. Our method to create human-annotated rankings can be used to create datasets for different domains that would, over time, allow for such an extensive analysis.

CONCLUSION

In this paper, we showed why currently available datasets are not suitable for testing the subjective accuracy of similarity measures and demonstrated how human-annotated rankings can be created using crowdsourcing tasks. We published a freely available dataset comprising 8 different human-annotated rankings including the query pattern and 15 subsequences per query. This dataset can be used to optimize new and existing algorithms and serves as starting point for further research in this area. Furthermore, we collected 112 different query patterns from different users, analyzed their meaning and characteristics and proposed future ideas for a visual vocabulary for time series queries.

All queries and human-annotated rankings are available on `http://cs.brown.edu/~peichmann/timeseries/`

ACKNOWLEDGMENTS

The authors wish to thank Bob Zeleznik, Joseph La Viola and Andries van Dam for their comments and suggestions, Yueguo Chen and Eamon Keogh for the source code as well all our participants and Mechanical Turk workers for their time and their invaluable feedback.

REFERENCES

1. André-Jönsson, H., and Badal, D. Z. Using signature files for querying time-series data. In *Principles of Data Mining and Knowledge Discovery*. Springer, 1997, 211–220.
2. Batista, G. E., Wang, X., and Keogh, E. J. A complexity-invariant distance measure for time series.
3. Chen, L., Özsu, M. T., and Oria, V. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, ACM (2005), 491–502.
4. Chen, Y., Nascimento, M. A., Ooi, B. C., and Tung, A. Spade: On shape-based pattern detection in streaming time series. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, IEEE (2007), 786–795.
5. De Berg, M., Van Kreveld, M., Overmars, M., and Schwarzkopf, O. C. *Computational geometry*. Springer, 2000.
6. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment 1*, 2 (2008), 1542–1552.
7. Eitz, M., Hays, J., and Alexa, M. How do humans sketch objects? *ACM Trans. Graph. 31*, 4 (2012), 44.
8. Faloutsos, C., Ranganathan, M., and Manolopoulos, Y. *Fast subsequence matching in time-series databases*, vol. 23. ACM, 1994.
9. Fu, T.-c., Chung, F.-l., Luk, R., and Ng, C.-m. Stock time series pattern matching: Template-based vs.

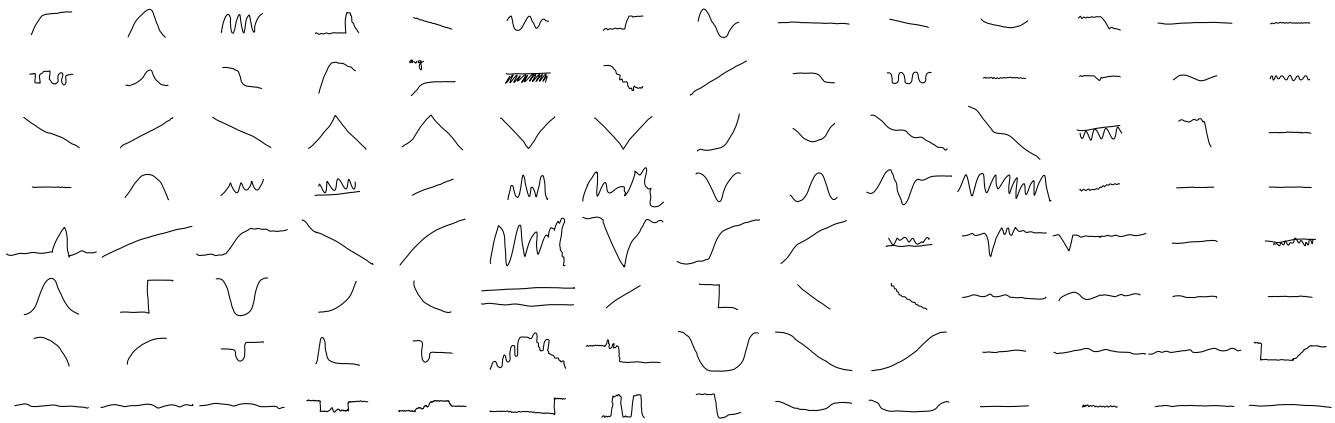

Figure 8. 112 Pen-drawn query patterns collected from 6 different users.

rule-based approaches. *Engineering Applications of Artificial Intelligence 20*, 3 (2007), 347–364.

10. Gregory, M., and Shneiderman, B. Shape identification in temporal data sets. 305–321.
11. Hochheiser, H., and Shneiderman, B. Dynamic query tools for time series data sets: timebox widgets for interactive exploration. *Information Visualization 3*, 1 (2004), 1–18.
12. Hüllermeier, E., Fürnkranz, J., Cheng, W., and Brinker, K. Label ranking by learning pairwise preferences. *Artificial Intelligence 172*, 16 (2008), 1897–1916.
13. Keogh, E., and Kasetty, S. On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Mining and knowledge discovery 7*, 4 (2003), 349–371.
14. Keogh, E. J., and Pazzani, M. J. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In *KDD*, vol. 98 (1998), 239–243.
15. Keogh, E. J., and Pazzani, M. J. Relevance feedback retrieval of time series data. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, ACM (1999), 183–190.
16. Keogh, E., Z. Q. H. B. H. Y. X. X. W. L. . R. *The UCR Time Series Classification/Clustering Homepage*, 2011. `www.cs.ucr.edu/~eamonn/time_series_data/`.
17. Mitsa, T. *Temporal data mining*. CRC Press, 2010.
18. Morse, M. D., and Patel, J. M. An efficient and accurate method for evaluating time series similarity. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, ACM (2007), 569–580.
19. Palpanas, T., Vlachos, M., Keogh, E., Gunopulos, D., and Truppel, W. Online amnesic approximation of streaming time series. In *Data Engineering, 2004. Proceedings. 20th International Conference on*, 339–349.
20. Papapetrou, P., Athitsos, V., Potamias, M., Kollios, G., and Gunopulos, D. Embedding-based subsequence matching in time-series databases. *ACM Transactions on Database Systems (TODS) 36*, 3 (2011), 17.
21. Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., and Keogh, E. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM (2012), 262–270.
22. Ratanamahatana, C., Keogh, E., Bagnall, A. J., and Lonardi, S. A novel bit level time series representation with implication of similarity search and clustering. In *Advances in Knowledge Discovery and Data Mining*. Springer, 2005, 771–777.
23. Vlachos, M., Hadjieleftheriou, M., Gunopulos, D., and Keogh, E. Indexing multidimensional time-series. *The VLDB JournalThe International Journal on Very Large Data Bases 15*, 1 (2006), 1–20.
24. Vlachos, M., Kollios, G., and Gunopulos, D. Discovering similar multidimensional trajectories. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, IEEE (2002), 673–684.
25. Wan, X., Yang, J., and Xiao, J. Towards a unified approach to document similarity search using manifold-ranking of blocks. *Information Processing & Management 44*, 3 (2008), 1032–1048.
26. Wattenberg, M. Sketching a graph to query a time-series database. In *CHI'01 Extended Abstracts on Human factors in Computing Systems*, ACM (2001), 381–382.

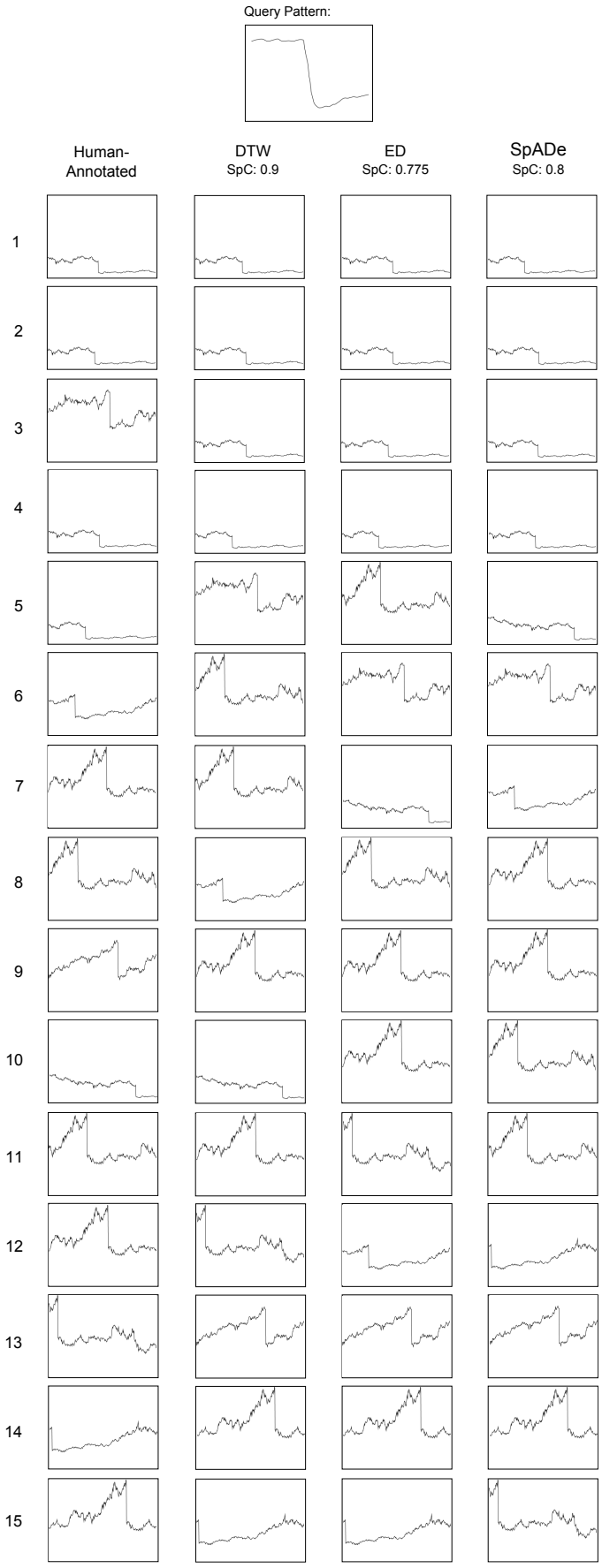

Figure 9. A human-annotated ranking for query pattern 2 and three different rankings produced by DTW, ED, and SpAde similarity measure using PIP sampling. The algorithms produce reasonable results according to the Spearman coefficient (SpC). Note that even the crowd sometimes fails to agree upon a most similiar shape (see rank 3 in the human-annotated ranking).