Classifying Time-Series Data

Matthew Faw

May 2, 2016

1 Problem Description

To extract useful information from data, it is often desirable to identify commonalities and trends. To identify such trends exhibited in data, a notion of closeness of data is often necessary. While many natural and intuitive measures of distance between points exist (Euclidean distance, for example), measuring the distance between two curves is much less natural. In order to classify and cluster time-varying data, however, defining distance between curves is essential.

Time series data is formally defined as a sequence of pairs $T = \{(p_1, t_1), (p_2, t_2), \ldots, (p_n, t_n)\}$, where $t_1 < t_2 < \ldots < t_n$ and each p_i, $1 \leq i \leq n$ is a point in d-dimensional space, and each t_i is the corresponding time stamp at which p_i occurs [1]. Examples of such data are numerous and growing, and include time series of handwritten signatures, which have been used for biometric authentication systems[2], and GPS streams of taxi trajectories, which have been used to aid maintenance of road maps.

The purpose of this work is to investigate the performance of trajectory classification under both well-known and recently proposed distance measures. We will examine data sets that exhibit variations in sampling rates and presence of outliers in order to better understand the conditions under which each distance measure performs well or poorly. Using the information gained through this study, we will offer suggestions for future work.

2 Background

2.1 Distance Measurement Algorithms

We will begin by discussing some of the most common time series distance measures, as well as recent developments and improvements upon these algorithms.

2.1.1 Frechet Distance

One commonly used measure of similarity between between curves is the Frechet distance [3]. Let f and g be two functions mapping from homeomorphic parameter spaces A and B to d-dimensional space. The Frechet distance between f and g is defined to be:

$$\delta_F(f, g) = \inf_{\sigma: A \to B} \sup_{x \in A} \|f(x) - g(\sigma(x))\| \tag{1}$$

Intuitively, the Frechet distance can be though of as follows: suppose a person is walking along a trajectory, and his dog is walking along another trajectory, where the person holds a leash connected to the dog. Neither the person nor the dog may backtrack along their respective paths. The Frechet distance is the minimum leash length that can allow for the person and dog to walk along the curves as described. The discrete Frechet distance is used when we do not have an entire trajectory, only sample points along the trajectory. We note that since there may be a large number of matchings between trajectories which yield the optimal Frechet distance, and thus Frechet distance may not always provide insight into trajectory similarity.

We may also consider the discretized version of this problem. Define a *coupling* L between polygonal curves P and Q as a sequence $(p_{a1}, q_{b1}), (p_{a2}, q_{b2}), \ldots, (p_{am}, q_{bm})$ of distinct pairs of vertices from the polygonal chains such that $a_1 = 1, b_1 = 1, a_m = |Q|, b_m = |Q|$, and for all $i = 1, \ldots, |Q|$ we have $a_{i+1} = a_i$ or

$a_{i+1} = a_i + 1$ and $b_{i+1} = b_i$ or $b_{i+1} = b_i + 1$. The length of the coupling L is the length of the longest link in L, $\|L\| = \max_{i=1,\ldots,m} |p_{ai} - q_{bi}|$. The *discrete Frechet Distance* is thus defined as:

$$\delta_{dF}(P, Q) = \min\{\|L\| | L \text{is a coupling between } P \text{ and } Q\}. \tag{2}$$

The dynamic programming algorithm used to efficiently compute the discrete Frechet distance is given in [4].

2.1.2 Dynamic Time Warping

Since a pair of trajectories may have a large number of possible matchings, it may be desirable to minimize the average distance of the leash from Frechet distance. This is precisely what Dynamic Time warping does. In this algorithm, the recurrence is:

$$DTW(x_i, y_j) = dist(x_i, y_j) + \min DTW(x_{i-1}, y_j), DTW(x_i, y_{j-1}), DTW(x_{i-1}, y_{j-1}) \tag{3}$$

Using dynamic programming, this recurrence can be solved in $O(mn)$ time, where m is the length of x and n is the length of y [5]. Initially coming from the spoken word recognition field, Dynamic time warping is a commonly-used distance measure for time series, and is convenient because, unlike Euclidean distance, it can be applied to sequences of different lengths [6].

2.1.3 Longest Common Subsequence

Another method for determining the distance between two multidimensional trajectories is described in [7]. It is based on the longest common subsequence problem. The problem is: given an integer δ and a real number $0 < \epsilon < 1$,

$$LCSS_{\delta,\epsilon}(P, Q) = \begin{cases} 0, & \text{if P or Q is empty.} \\ 1 + LCSS_{\delta,\epsilon}(P[1:(m-1)], Q[1:(n-1)]), & \forall i = 1 \ldots \text{dimension}, |p_{x_i,m} - q_{x_i,n}| < \epsilon, |n-m| < \delta. \\ \max\{LCSS_{\delta,\epsilon}(P[1:(m-1)], Q), \ldots & \\ \quad LCSS_{\delta,\epsilon}(P[1:(m-1)], Q[1:(n-1)]), \ldots & \\ \quad LCSS_{\delta,\epsilon}(P, Q[1:(n-1)])\}, & \text{otherwise} \end{cases}$$

We may then define the distance between P and Q to be:

$$D(\delta, \epsilon, P, Q) = 1 - \frac{LCSS_{\delta,\epsilon}(P, Q)}{\min m, n}.$$

Note that this method claims to be robust to outliers, since it allows some elements to go unmatched.

2.1.4 Assignment

In [8], the authors introduce a new method for scoring assignments which encompasses dynamic time warping, sequence alignment, and others, while avoiding their drawbacks when outliers or irregular sampling are present. The authors begin by defining an **assignment** for two sequences P and Q as a pair of functions $\alpha : P \to Q \cup \{\perp\}$, $\beta : Q \to P \cup \{\perp\}$ for the points of P and Q respectively. If $\alpha(p_i) = \perp$(or $\beta(q_j) = \perp$, then p_i (or q_j) is called a **gap point**. A maximal contiguous sequence of gap points in P or Q is called a **gap**. Because their matching algorithm allows asymmetric assignments, their model can better adapt to trajectories with different sampling rates. Let E be a the set of edges connecting P and Q, and let $\Gamma(E)$ be the set of gaps in P and Q. Then the score of E is

$$\sigma(P, Q; E) = \sum_{(u,v)\in E} \frac{1}{\lambda + \|u - v\|^2} + \sum_{g\in\Gamma(E)} (\Theta + \Delta|g|) \tag{4}$$

Where $\Theta < 0, \Delta > 0, \lambda > 0$ are parameters. Rewriting in terms of the definitions of α and β,

$$\sigma(P, Q; \alpha, \beta) = \sum_{p_i\in P, \alpha(p_i)\neq\perp} \frac{1}{\lambda + \|p_i - \alpha(p_i)\|^2} + \sum_{q_j\in Q, \alpha(q_j)\neq\perp} \frac{1}{\lambda + \|q_j - \beta(q_j)\|^2} + \sum_{g\in\Gamma(\alpha,\beta)} (\Theta + \Delta|g|) \tag{5}$$

The similarity between P and Q is thus defined as

$$\sigma(P,Q) = \max_{\alpha,\beta} \sigma(P,Q,\alpha,\beta) \tag{6}$$

This problem can be solved using dynamic programming using the methods introduced in [8].

2.1.5 Parametric Derivative Time warping

[9] introduces a new shape-based similarity measure called parametric derivative dynamic time warping for multivariate time series data. This work allows the expansion of dynamic time warping into multiple dimensions. The equation for this measure is

$$DD_{DTW}(X,Y) = (1-\alpha)DTW(X,Y) + \alpha DDTW(X,Y) \tag{7}$$

where α is a parameter and $DDTW$ is the DTW of the numerical derivatives of X and Y. Although α must be determined through a time-consuming cross-validation step, the parametric approach used in this method allow combining the advantages of DTW and DDTW while avoiding their drawbacks by allowing tuning error of the DTW and DDTW contributors.

2.1.6 Move-Split-Merge Metric

The main idea behind the Move-Split-Merge metric is to define a set of operations that can turn any time series into any other time series. These three fundamental operation are *move,* which changes a single point of the time series, *split*, which splits a single point of the time series, and *merge*, which merges to points with the same value to a single point with the same value. The similarity measure aims for robustness to misalignment, metricity, translation invariance, treating all values equally, and quadratic time complexity. The recurrence formulation is:

$$Cost(i,j) = \min\{Cost(i-1,j-1) + |x_i - y_j|, Cost(i-1,j) + C(x_i, x_{i-1}, y_1), Cost(i,j-1) + C(y_j, x_i, y_{j-1})\}$$

$$C(x_i, x_{i-1}, y_j) = \begin{cases} c, & \text{if } x_{i-1} \le x_i \le y_j \text{ or } x_{i-1} \ge x_i \ge y_j. \\ c + \min(|x_i - x_{i-1}|, |x_i - y_j|), & \text{otherwise.} \end{cases}$$

The primary distinction between MSM and edit distance is they way insertions and deletions are handled. The MSM cost model aims not for all insertions and deletions to cost the same as in edit distance, but to depend both on that value and the adjacent value [10].

2.2 Classification Algorithm Design

We will examine classification performance under these three methods of measuring the distance between trajectories. Thus, the ideal classifier for trajectories is one which we can easily modify from measuring the distance between points (in the traditional setting) to measuring the distance between curves (in this setting). A natural choice is the k-Nearest neighbors algorithm [11], which, for every trajectory in the test set, finds the k closest neighbors in the training set, and measures performance based on how many of those curves are in the class of interest. The decision statistic is calculated by:

$$\lambda(x) = \frac{\text{Number of trajectories in class of interest in k nearest}}{k}$$

Due to the slow speeds at which kNN runs when measuring distance between trajectories, k was restricted to 1. Many sources suggest that this is the optimal choice of k [6, 12].

Another classifier, the Distance Likelihood Ratio Test [13], is also a good candidate for classification. This classifier is conceptually very similar to kNN, but it additionally takes into account the distance to the kth neighbor. The decision statistic is calculated by

$$\lambda(x) = \ln(\frac{n_0}{n_1}) + D[\ln \Delta_{k0} - \ln \Delta_{k1}],$$

where n_0 is the number of data points in H_0, n_1 is the number of data points in H_1, D is the dimensionality of the data, Δ_{k0} is the distance from each test point to the kth closest training point in H_0, and similarly Δ_{k1} is the distance from each test point to the kth closest training point in H_1.

In order to measure classification performance, I obtained $P_D(\beta)$, the probability of detection, and $P_{FA}(\beta)$, the probability of false alarm, given $\lambda \geq \beta$, from each classification. These probabilities are calculated by:

$$P_D(\beta) = P(\text{Decide } H_1 \text{ — data from } H_1, \beta) = \frac{\text{Number of class-1 test points with decision statistic at least } \beta}{\text{total number class-1 test points}}$$

$$P_{FA}(\beta) = P(\text{Decide } H_0 \text{ — data from } H_0, \beta) = \frac{\text{Number of class-0 test points with decision statistic at least } \beta}{\text{total number class-0 test points}}$$

By calculating P_D and P_{FA}, we may generate ROC curves and calculate area under the ROC curve in order to measure performance.

3 Methods

3.1 Data Description

All of the data used in this project was obtained from [14]. [6] provides a description of all of the data. Of this data, I chose to use Gun-Point and MoteStrain. Gun-Point was chosen due to the simplicity and regularity of the curves. This data provides a nice control, as there are few outliers. This data set is made up of two classes, GunDraw and GunPoint. The data is created from a recording peoples' right hands as they draw or pretend to draw (respectively) a gun. In the GunDraw class, there is a small peak that distinguishes the two classes. Figure 1 shows a plot of one time series from each class.

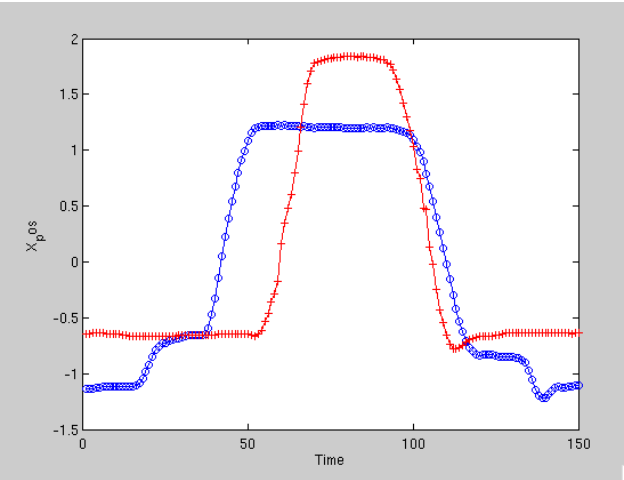

Figure 1: Z-Normalized Gun trajectories

MoteStrain was chosen because it has very sharp dropouts. This will allow us to determine how various classifiers perform under noisy data. This data has two classes, and contains sensor measurements of humidity and temperature. Figure 2 shows one time series from each class.

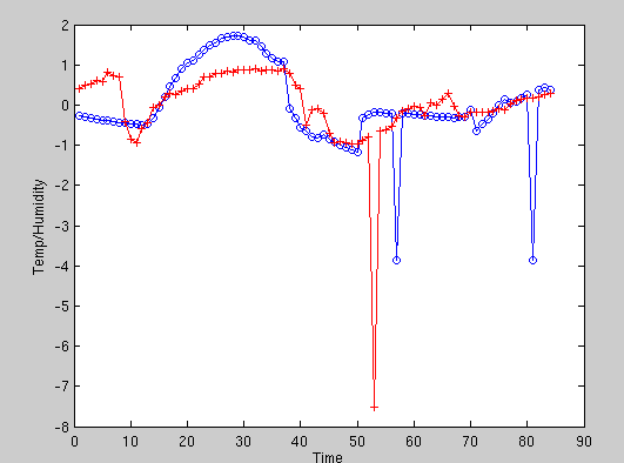

Figure 2: Z-Normalized Mote Strain trajectories

3.2 Pre-processing

Only minimal preprocessing was necessary for the data. The each trajectory, as provided, was individually normalized using the zero-mean unit-variance equation:

$$\hat{x} = \frac{x - \mu}{\sigma} \tag{8}$$

where μ is the mean of the trajectory and σ is the standard deviation. This normalization is essential, as, by [15], scaling and offset can very negatively affect classification performance.

Because every trajectory within a each dataset is the same length, preprocessing on the data must be done to observe the effects of sampling on distance calculations. For a simple time series (i.e. one that can be described by only a few data points), we desire that the distance calculation shows little fluctuation. Thus, we perform additional preprocessing on two simple trajectories (trajectories from the Gun dataset). We decrease the number of points in the first sequence by half until we reach a unit-length sequence. At each step, we compare the distance between the two curves.

In order to reduce computational load, each dataset tested was reduced to size of twelve trajectories.

4 Results

I ran the 1-nearest neighbor algorithm and distance likelihood ratio tests on the gun dataset. $k = 1$ was chosen based on suggestions in literature [6, 12] that $k = 1$ is hard to beat. The 3-folds cross validated ROC results for each distance algorithm for 1NN are shown in Figures 4,6,8,9, and 11. The 3-folds cross validated ROC results for each distance algorithm for DLRT are shown in Figures 5,7,10, and 12.

I ran the 1NN and distance likelihood ratio test for $k = 1$ on the the MoteStrain dataset. The 3-folds cross validated ROC curves for each of the distance algorithms for 1NN are shown in Figures 14,16,19, and 21. The 3-folds cross validated ROC curves for each of the distance algorithms for DLRT are shown in Figures 13,15,17,18, and 20.

Finally, I took the first two time series from the Gun dataset, used each distance algorithm to determine the initial distance, and reduced the number of points in the first time series by half at each iteration. The plot of distance versus iteration (number of halving steps performed) is shown in Figure 3.

5 Discussion

First we discuss the 1NN data. Figures 4 and 6 both perform very well on the gun dataset, with their ROC plots relatively close to the upper left-hand corner of the page. The longest common subsequence

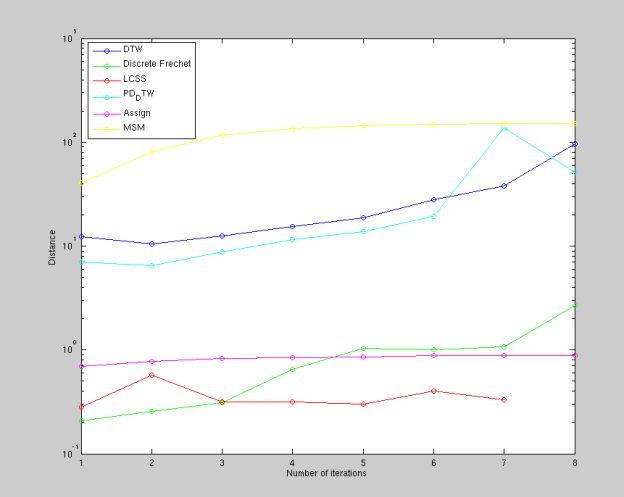

Figure 3: Distance versus iteration for each distance algorithm

(8), however, performs surprisingly poorly. Because there are both δ and ϵ parameters to set in the LCSS distance, it is likely that a poor parameter choice was made so that performance was low. The assignment distance (Figure 9) [8] and parametric derivative dynamic time warping distance (Figure 11) both perform similarly decently, although surprisingly not as well as DTW or Discrete Frechet. Perhaps this less-than-optimal performance is due to poor parameter tuning. Indeed, the advanced parameter tuning techniques discussed in [8] and [9] were not invoked here; instead, parameters were chosen based on informed guesses of the values based around the standard deviation of the data. When this data was tested using DLRT, the trends were overall the same, although the overall classification performance was lower (i.e. less area under the ROC curve).

First, we discuss the DLRT data. Figure 13 shows DTW distance again has relatively strong performance amongst the Mote Strain dataset ROC curves, with the curve lying somewhat close to the upper left hand corner. The ROC curve for LCSS distance shows very similar performance. Thus, it appears that both DTW and LCSS hold up well against the fall-outs in the dataset. Based upon the observations of [8], this is expected, as LCSS is designed to perform well under outliers. Figure 15 shows very poor performance of Frechet distance under the MoteStrain dataset. This implies that the discrete Frechet distance is quite sensitive to outliers. Since the discrete Frechet distance algorithm has to have a "long enough leash to walk the dog along the curve", this sensitivity to outliers seems intuitive. The Assignment algorithm from [8] performs relatively well (as seen in Figure 18). Although it intercepts the p_D axis at a lower point that DTW or LCSS, the overall curve seems to be pulled slightly more towards the upper left hand corner of the graph. Because the parameters for the Assignment algorithm were not finely tuned as they are supposed to be, it is likely that this algorithm could perform better, given better parameters. Figure 20 shows the performance of parametric derivative DTW is slightly lower than DTW or LCSS. Although the curve intercepts the p_D axis at a similar location to both the DTW and LCSS curves, there is a sharper knee in the pdDTW plot, showing weaker performance. It is possible that this algorithm, too, is under performing, since it's α parameter, as discussed earlier in the paper, was not properly tuned. We notice here that the 1NN data exhibits the same trends as the DLRT data. In all cases but DTW distance, the 1NN classifier seems to perform better.

Figure 3 shows that, as the number of points in the trajectory decreases, the Align algorithm tends to output the most stable distance output. The longest common subsequence, although mostly stable, does show some fluctuation. Both DTW and parametric derivative DTW show slight upward trends, indicating that decreased sampling decreases performance of distance calculation. From this plot, the Assignment algorithm shows the best stability to decreased sampling. Thus, it appears that, when dealing with datasets with low sample rates, the Assignment algorithm is ideal among the distance algorithms tested.

6 Related Work

An extensive amount of attention has been focused on *Map Matching*: a study focused on aligning a sequence of observed user positions with the road network on a digital map. The sampling rate is often relatively low (one sample every 2-5 minutes). Thus, traditional time series distance measures fall apart in these settings. In order to compensate for this, researchers have developed means to exploit both spacial and temporal information (including posted suggested road speeds) to increase accuracy [16]. Additionally, Hidden Markov models have been introduced to improve online traffic sensing [17].

In the deep learning community, significant progress has been made in time series classification as well. One research group introduced a method for time series classification using multi-channel deep convolutional neural networks. This model has an incredibly efficient runtime (better than state of the art), while also being competitive in accuracy of classification [18].

In order to compensate for slow runtimes of many time series similarity measures, one group introduced a time-series classification method which involves Fast Matrix Factroization and SVM classifiers. While showing similar classification performance, this method removes cubic orders of classification time complexity, running worst case $O(n)$ time [19].

7 Suggested Work

The performance of the distance algorithm presented in [8] suggests promising results for classification under this distance measure. This work also suggests exploration of another type of classification algorithm. Work presented in [20] discusses the potential benefits of *shapelets* in time series classification. A shapelet is a time-series subsequence that allows for time series classification based on local, phase-independent similarities in shape. This paper presents a shapelet search algorithm. I propose that the segments generated in the segmentation algorithm of [8] be used as the shapelets. Generation of these segments takes $O(n^2)$ time, where n is the total number of trajectories. Once these segments have been obtained, a quality measure such as the F-statistic will be used to determine the best k shapelets. The choice of quality measure can dramatically alter the runtime of the algorithm–[20] showed that the F statistic performs well in practice. As in [20], the k shapelets can be reduced either by ignoring shapelets below a cut-off point or by clustering the shapelets. A new transformed dataset is then created, so that each attribute represents a shapelet, and the value of the attribute is the distance between the shapelet and the original series. As a result of this transformation, shapelet finding is disassociated from classification. Thus, transformed data can be used in conjunction with any classifier.

8 Conclusions

Overall, none of the observed results seem to discredit the acclaimed good performance of 1-nearest neighbor algorithm using dynamic time warping distance. However, the results of both the Assignment algorithm and Parametric derivative DTW are somewhat inconclusive, since the parameters were not as tightly tuned as specified in [8] and [9]. Further experiments would have to be performed to obtain more conclusive evidence of their superiority in performance.

The assignment algorithm seems to respond most ideally to changes in sampling rate. Thus, in low sample rate datasets, the Assignment algorithm may likely perform well. This area should be further explored.

The 1NN classifier was tested side-by-side with the DLRT classifier. In almost every case, the 1NN classifier seems to have overall better classification perform than DLRT. However, both classifiers demonstrated the same trends in the data. Thus, neither classifier seems to be "misbehaving." A more thorough comparative into performance of various classifiers under a range of distance measurements would be an intersting study (though one that should be conducted on a cluster of computers).

On the note of performance of these algorithms, practically all are based dynamic programming, and thus distance computation has quadratic runtime. Sub-quadratic time approximation algorithms have been proposed for Dynamic Time Warping and Edit-distance [21]. Perhaps similar approximation algorithms could be applied to the algorithm proposed in [8] to make using such algorithms more scalable.

References

[1] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: Experimental comparison of representations and distance measures. *Proc. VLDB Endow.*, 1(2):1542–1552, August 2008.

[2] Y. Manabe and B. Chakraborty. Identity detection from on-line handwriting time series. In *Soft Computing in Industrial Applications, 2008. SMCia '08. IEEE Conference on*, pages 365–370, June 2008.

[3] Kevin Buchin, Maike Buchin, and Carola Wenk. Computing the fréchet distance between simple polygons. *Comput. Geom. Theory Appl.*, 41(1-2):2–20, October 2008.

[4] Technische Universität Wien, Thomas Eiter, Thomas Eiter, Heikki Mannila, and Heikki Mannila. Computing discrete fréchet distance. Technical report, 1994.

[5] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowl. Inf. Syst.*, 7(3):358–386, March 2005.

[6] A distance based time series classification framework. *Inf. Syst.*, 51(C):27–42, July 2015.

[7] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 673–684, 2002.

[8] Swaminathan Sankararaman, Pankaj K. Agarwal, Thomas Mølhave, Jiangwei Pan, and Arnold P. Boedihardjo. Model-driven matching and segmentation of trajectories. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL'13, pages 234–243, New York, NY, USA, 2013. ACM.

[9] Tomasz Gorecki and Maciej Luczak. Multivariate time series classification with parametric derivative dynamic time warping. *Expert Syst. Appl.*, 42(5):2305–2312, April 2015.

[10] A. Stefan, V. Athitsos, and G. Das. The move-split-merge metric for time series. *IEEE Transactions on Knowledge and Data Engineering*, 25(6):1425–1438, June 2013.

[11] P. Viswanath and T. Hitendra Sarma. An improvement to k-nearest neighbor classifier. In *Recent Advances in Intelligent Computational Systems (RAICS), 2011 IEEE*, pages 227–231, Sept 2011.

[12] Xiaopeng Xi, Eamonn Keogh, Christian Shelton, Li Wei, and Chotirat Ann Ratanamahatana. Fast time series classification using numerosity reduction. In *Proceedings of the Twenty-Third International Conference on Machine Learning*, pages 1033–1040, 2006.

[13] J. J. Remus, K. D. Morton, P. A. Torrione, S. L. Tantum, and L. M. Collins. Comparison of a distance-based likelihood ratio test and k-nearest neighbor classification methods. In *2008 IEEE Workshop on Machine Learning for Signal Processing*, pages 362–367, Oct 2008.

[14] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive, July 2015.

[15] Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn Keogh. Addressing big data time series: Mining trillions of time series subsequences under dynamic time warping. *ACM Trans. Knowl. Discov. Data*, 7(3):10:1–10:31, September 2013.

[16] Yin Lou, Chengyang Zhang, Yu Zheng, Xing Xie, Wei Wang, and Yan Huang. Map-matching for low-sampling-rate gps trajectories. In *ACM SIGSPATIAL GIS 2009*. Association for Computing Machinery, Inc., November 2009.

[17] C. Y. Goh, J. Dauwels, N. Mitrovic, M. T. Asif, A. Oran, and P. Jaillet. Online map-matching based on hidden markov model for real-time traffic sensing applications. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pages 776–781, Sept 2012.

[18] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J. Leon Zhao. *Web-Age Information Management: 15th International Conference, WAIM 2014, Macau, China, June 16-18, 2014. Proceedings*, chapter Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks, pages 298–310. Springer International Publishing, Cham, 2014.

[19] Josif Grabocka, Erind Bedalli, and Lars Schmidt-Thieme. *ICT Innovations 2012: Secure and Intelligent Systems*, chapter Efficient Classification of Long Time-Series, pages 47–57. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[20] Jon Hills, Jason Lines, Edgaras Baranauskas, James Mapp, and Anthony Bagnall. Classification of time series by shapelet transformation. *Data Min. Knowl. Discov.*, 28(4):851–881, July 2014.

[21] Pankaj K. Agarwal, Kyle Fox, Jiangwei Pan, and Rex Ying. Approximating dynamic time warping and edit distance for a pair of point sequences. *CoRR*, abs/1512.01876, 2015.

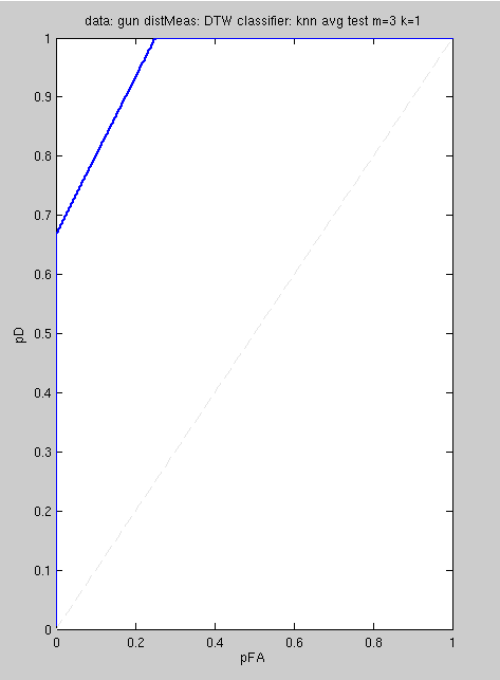

Figure 4: 3-Folds cross validated test ROC, gun dataset, dtw distance, 1NN

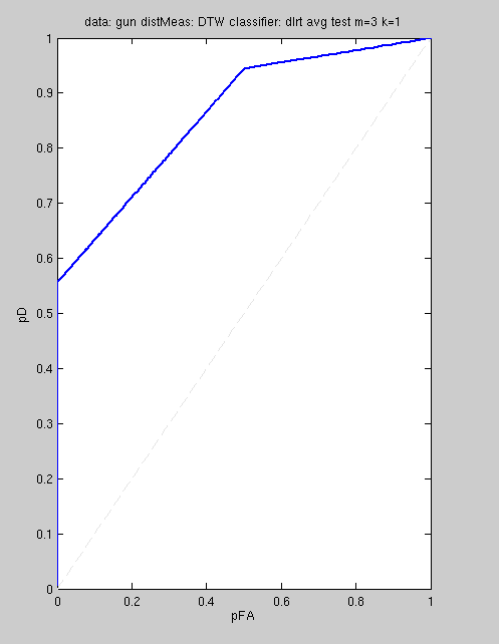

Figure 5: 3-Folds cross validated test ROC, gun dataset, dtw distance, DLRT, k=1

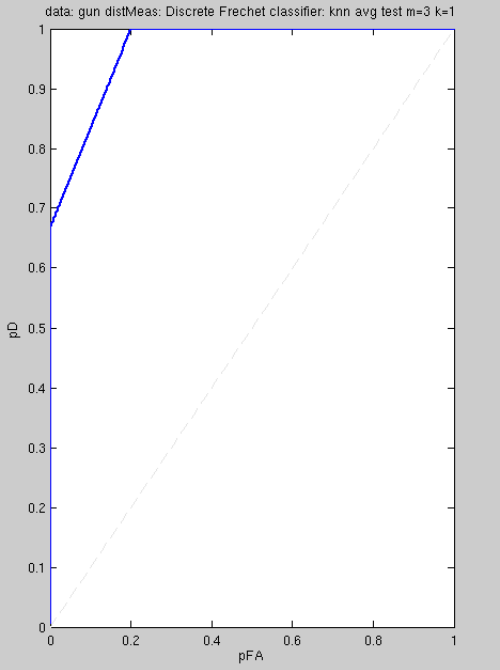

Figure 6: 3-Folds cross validated test ROC, gun dataset, Dicrete Frechet distance

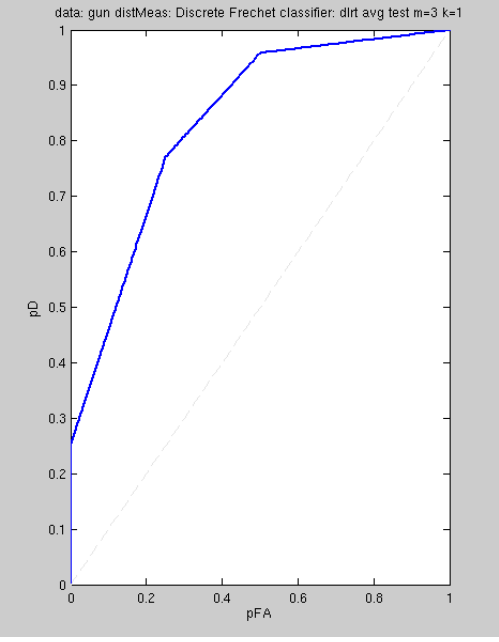

Figure 7: 3-Folds cross validated test ROC, gun dataset, discrete frechet distance, DLRT, k=1

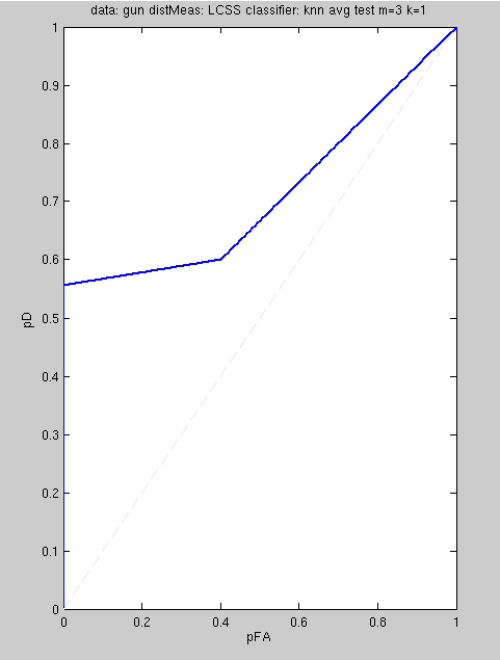

Figure 8: 3-Folds cross validated test ROC, gun dataset, LCSS distance

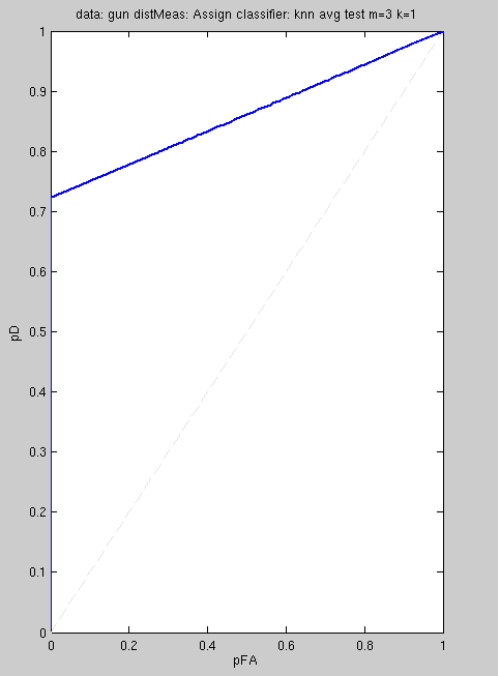

Figure 9: 3-Folds cross validated test ROC, gun dataset, Assignment distance

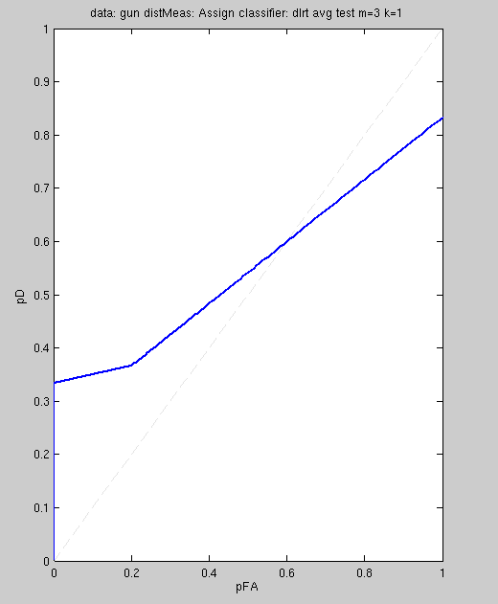

Figure 10: 3-Folds cross validated test ROC, gun dataset, Assignment distance, DLRT, k=1

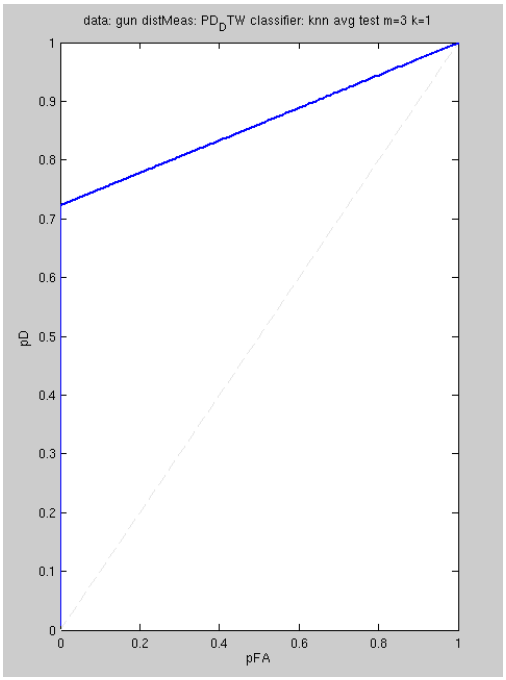

Figure 11: 3-Folds cross validated test ROC, gun dataset, parametric derivative dtw distance

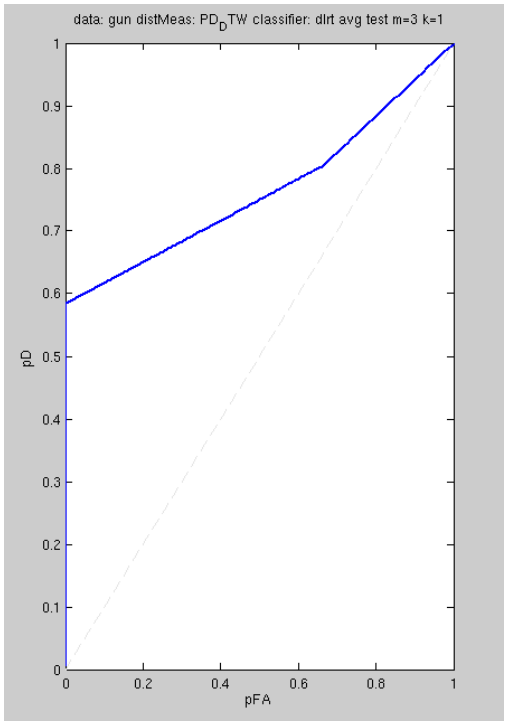

Figure 12: 3-Folds cross validated test ROC, gun dataset, parametric derivative dtw distance, DLRT, k=1

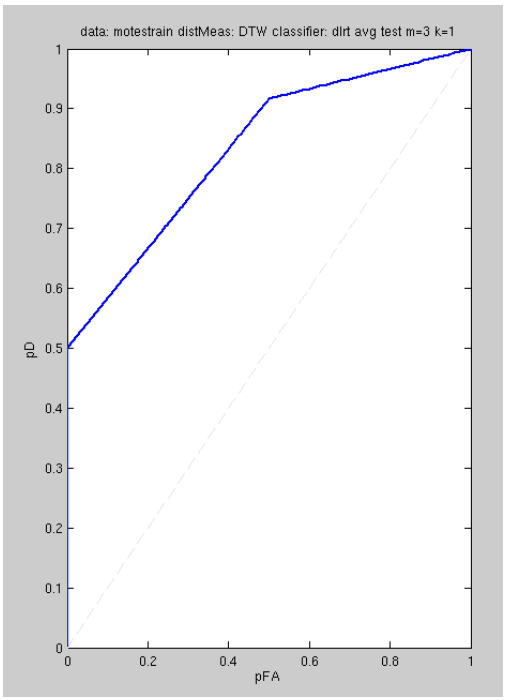

Figure 13: 3-Folds cross validated test ROC, Mote Strain dataset, dtw distance, DLRT

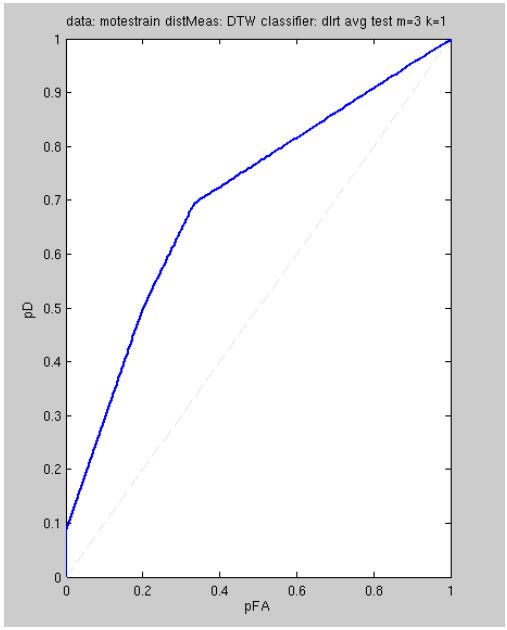

Figure 14: 3-Folds cross validated test ROC, Mote Strain dataset, dtw distance, 1NN

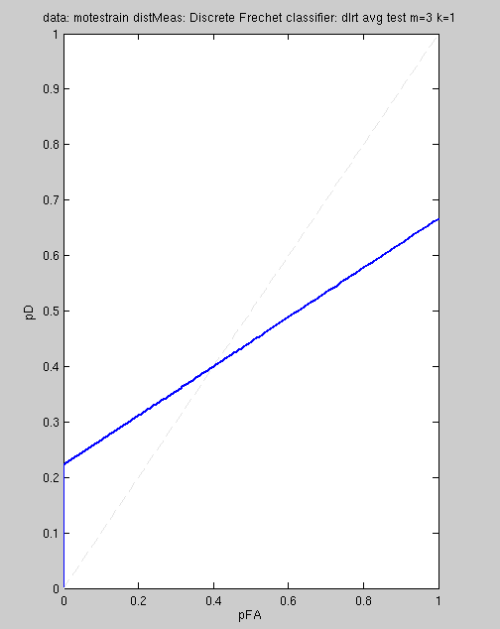

Figure 15: 3-Folds cross validated test ROC, Mote Strain dataset, discrete Frechet distance, DLRT

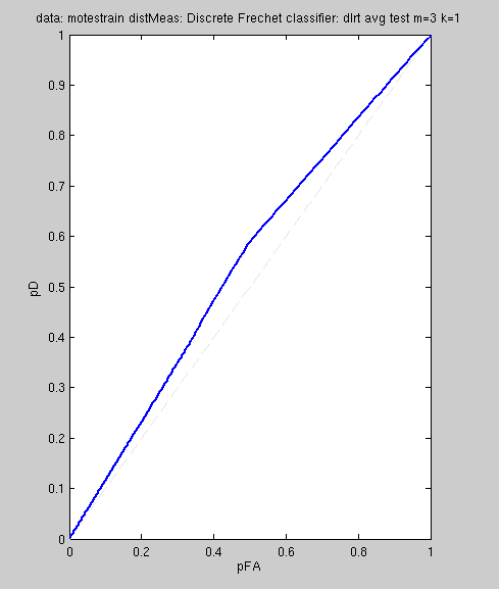

Figure 16: 3-Folds cross validated test ROC, Mote Strain dataset, discrete Frechet distance, 1NN

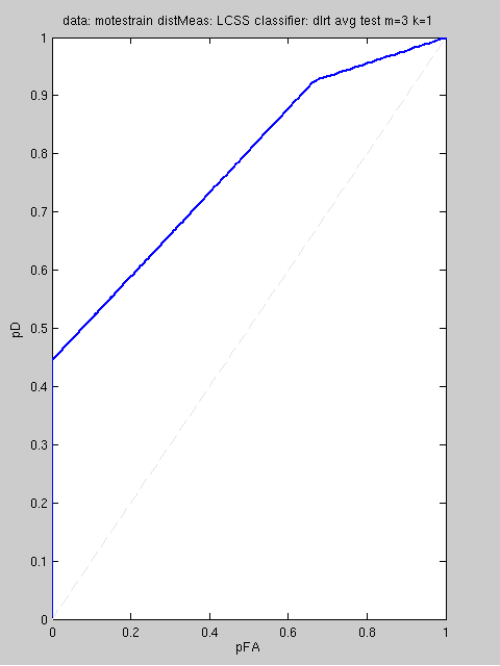

Figure 17: 3-Folds cross validated test ROC, Mote Strain dataset, LCSS distance, DLRT

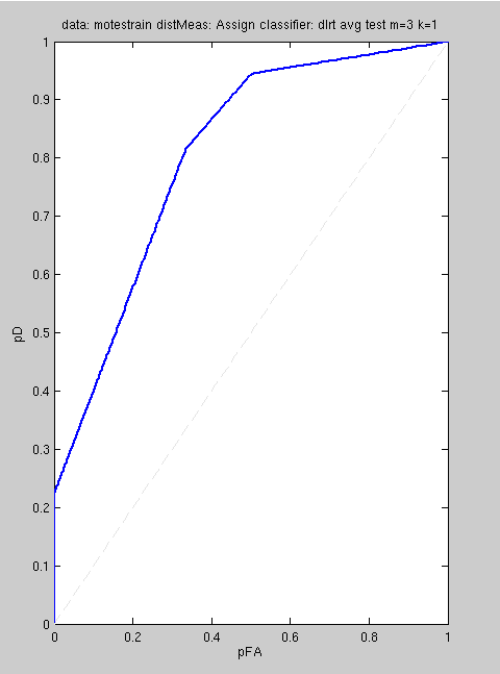

Figure 18: 3-Folds cross validated test ROC, Mote Strain dataset, Assignment distance. DLRT

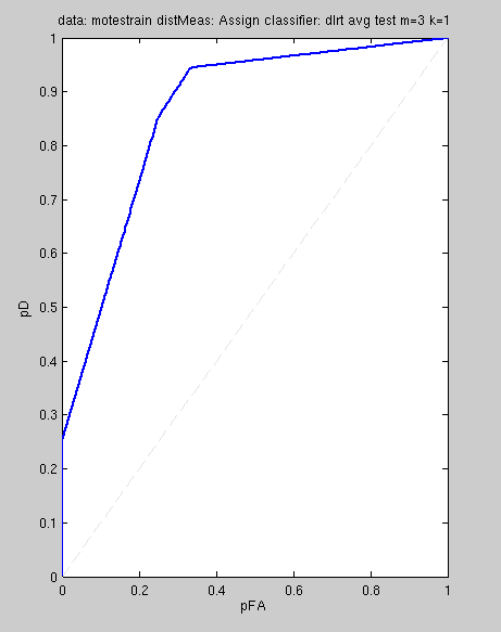

Figure 19: 3-Folds cross validated test ROC, Mote Strain dataset, Assignment distance, 1NN

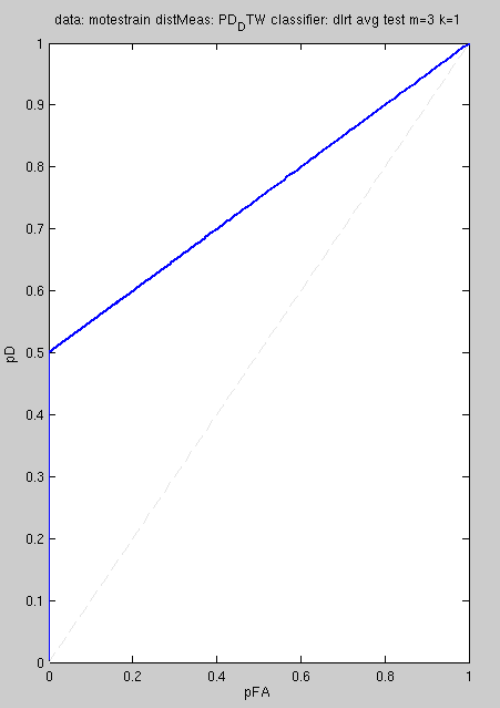

Figure 20: 3-Folds cross validated test ROC, Mote Strain dataset, parametric derivative dtw distance, DLRT

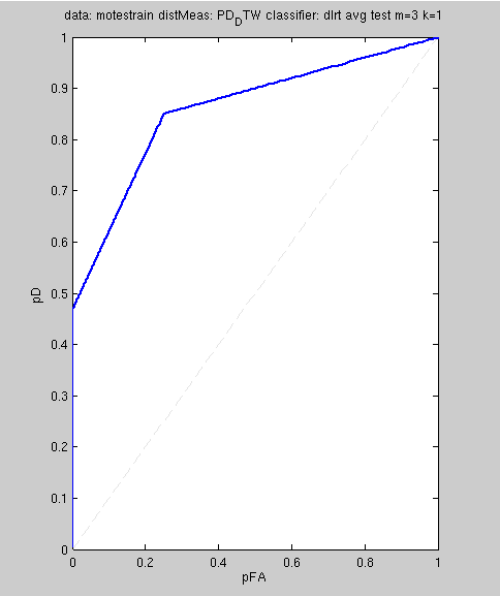

Figure 21: 3-Folds cross validated test ROC, Mote Strain dataset, parametric derivative dtw distance, 1NN