FastDTW is approximate and Generally Slower than the Algorithm it Approximates (Extended Abstract)

Renjie Wu
Computer Science & Engineering Department
University of California, Riverside
rwu034@ucr.edu

Eamonn J. Keogh
Computer Science & Engineering Department
University of California, Riverside
eamonn@cs.ucr.edu

***Abstract*—Many time series data mining problems can be solved with repeated use of distance measure. Examples of such tasks include similarity search, clustering, classification, anomaly detection and segmentation. For over two decades it has been known that the Dynamic Time Warping (DTW) distance measure is the best measure to use for most tasks, in most domains. Because the classic DTW algorithm has quadratic time complexity, many ideas have been introduced to reduce its amortized time, or to quickly approximate it. One of the most cited approximate approaches is FastDTW. The FastDTW algorithm has well over a thousand citations and has been explicitly used in several hundred research efforts. In this work, we make a surprising claim. In any realistic data mining application, the *approximate* FastDTW is much slower than the *exact* DTW. This fact clearly has implications for the community that uses this algorithm: allowing it to address much larger datasets, get exact results, and do so in less time.**

***Keywords*—Dynamic time warping, time series analysis, similarity measures, data mining**

I. Introduction

It has long been believed that the Dynamic Time Warping (DTW) distance measure is the best measure to use in many domains. DTW reports the distance of two time series after optimally aligning them. DTW is computed by finding the minimum cost path in distance matrix D of two time series X and Y, where $D(i,j) = (X[i] - Y[j])^2 + \min\{D(i\text{-}1,j\text{-}1), D(i\text{-}1,j), D(i,j\text{-}1)\}$. Since at least the 1970s, many practitioners have added *warping constraints*, denoted as w, to the allowable warping paths.

Because the classic DTW algorithm has a time complexity that is quadratic in the length of the sequences, many ideas have been introduced to reduce its amortized time [1], or to quickly approximate it [2]. One of the most cited approximate approaches is FastDTW [3]. FastDTW requires a parameter radius (r) and creates an approximation of classic (full) DTW by computing DTW on a downsampled version of the data, then iteratively projecting the solution discovered onto an upsampled version and refining it.

In this work, we make a surprising claim. In any realistic setting, FastDTW is actually *slower* than DTW. Every paper that we are aware of that uses FastDTW would have obtained faster results by using simple DTW. Moreover, these results would have been exact (by definition), not approximate.

II. Four Cases in Similarity Measurement

In this work, we use N to refer to the length of the time series being compared, W to refer to the *natural* amount of warping needed to align two random examples in a domain, given as a percentage of N. To be clear, we use FastDTW$_r$ for FastDTW with the radius of r and cDTW$_w$ to denote constrained DTW with the warping window width of w, also given as a percentage of N.

In Table I we can consider the following exhaustive and exclusive matrix of possible settings in which DTW can be used.

TABLE I. Four Settings in Which DTW Can Be Used

N gets larger →	W gets larger →	
	Case B Music performance, classical dance performance, seismic data	**Case D** <no obvious applications>
	Case A Heartbeats, gestures, signatures, golf swings, gene expressions, gait cycles, star-light-curves, sign language words or phrases, bird song	**Case C** Residential electrical power demand

A. Case A: Short N and Narrow W

For this case, cDTW is unambiguously faster. Moreover, the original authors echo this point and recommend cDTW [4]. In Fig. 1, we consider UWaveGestureLibraryAll dataset [5], which has exemplars of length 945, towards the long end of Case A.

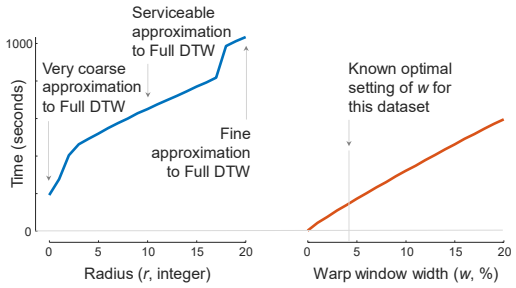

Fig. 1. A comparison of the time needed to compute all pairwise distances of the 896 training examples in UWaveGestureLibraryAll for r = 0 to 20 for FastDTW *(left)* and for w = 0 to 20% for cDTW *(right)*.

We can exactly compute cDTW$_{20}$ as fast as we can compute FastDTW$_{10}$, a serviceable approximation to Full DTW. Thus, this experiment provides forceful evidence that at least for Case A, FastDTW is slower than using cDTW. We believe that at least 99% of all uses of DTW in the literature fall into this case.

B. Case B: Long N and Narrow W

For Case B, imagine we align the exactly four-minute-long song with a live version. For classical music, various papers have suggested values such as W = 0.16% [6]. Let us assume that there can be mismatch of up to two seconds. Thus, we set w = 0.83%. Music processing typically uses Chroma Features,

which are normally sampled at 100Hz, thus we have a times series of length 24,000. We find that:

- $cDTW_{0.83}$ takes 45.6 milliseconds.
- $FastDTW_{10}$ takes 238.2 milliseconds.
- $FastDTW_{40}$ takes 350.9 milliseconds.

Thus, for Case B we find no evidence to use FastDTW.

C. Case C: Short N and Wide W

An example for Case C is residential electrical power demand, where N is reasonably short (450 datapoints), but W is a large fraction (153 datapoints) of this value, which gives us an estimate of $W = 34\%$. To be conservative, we will round up to 40%. In Fig. 2, we repeat the type of experiment shown in Fig. 1, but consider time series of length 450, and w from 0 to 40%.

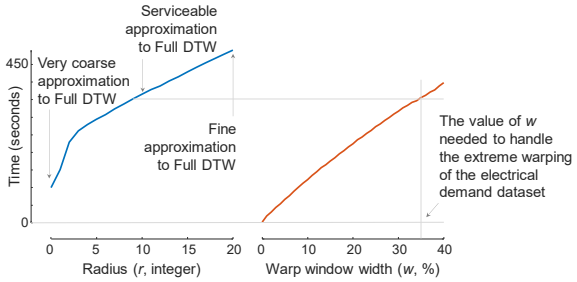

Fig. 2. Generalized experiment shown in Fig. 1 to consider w up to 40%.

Thus, for Case C we find no evidence of using FastDTW.

D. Case D: Long N and Wide W

Case D is the case emphasized by the original authors of the FastDTW paper as the best case for their algorithms. However, they did not show any real-world examples of such datasets, and we claim that there are no practical applications in Case D.

Nevertheless, for completeness we *do* test this case. As Fig. 3 shows, we created pairs of time series of length L seconds at 100Hz. In this domain, $W \approx 100\%$ and we must use $cDTW_{100}$.

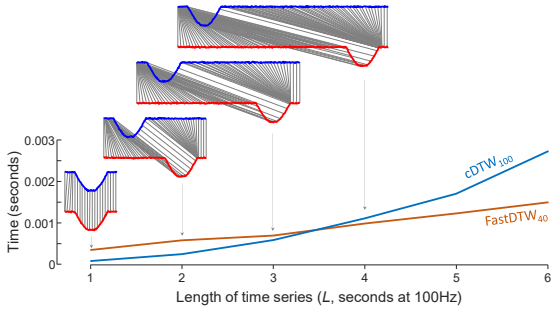

Fig. 3. *(top)* We created pairs of time series of length L seconds at 100Hz. *(bottom)* As we make L longer and longer, we find that when $L = 4$ ($N = 400$), $FastDTW_{40}$ finally becomes faster than Full DTW (or $cDTW_{100}$).

Thus, we have finally found a circumstance where $FastDTW_{40}$ is faster than $cDTW_{100}$. Note that at this breakeven point $FastDTW_{40}$ is an *approximation* to $cDTW_{100}$, so $cDTW_{100}$ is still preferable. However, as L grows well beyond the transition point, each user needs to consider the tradeoff between the time taken vs. utility of approximation.

III. When does FastDTW fail to Approximate Well?

We created three time series, and as shown in Table II, we measured their pairwise distances, using these distance matrices to create the dendrograms shown in Fig. 4.

TABLE II. The Distance Matrices for the Three Time Series Shown in Fig. 4 under Full DTW and $FastDTW_{20}$

Full DTW

	A	B	C
A	0	0.020	6.822
B		0	6.848
C			0

$FastDTW_{20}$

	A	B	C
A	0	31.24	6.822
B		0	6.848
C			0

Given unconstrained freedom to warp, A and B are almost identical, differing only by 0.02. However, $FastDTW_{20}$ finds them to be 31.24 apart. Using the error metric proposed in the original FastDTW paper [3], this is an error of 156,100%.

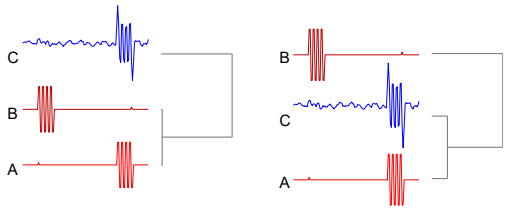

Fig. 4. A clustering of three time series under Full DTW *(left)* and under $FastDTW_{20}$ *(right)*.

IV. Conclusions

We have shown that the vast majority of the researchers that used FastDTW would have been better off simply using cDTW. They would have found simple cDTW to be both faster and to produce exact results. We discovered this issue because Salvador and Chan took enormous efforts to make their code available, to clearly explain their approach in their paper, and because they were incredibly responsive to the many questions we asked. We are extremely appreciative of their assistance.

Acknowledgment

We wish to thank Stan Salvador and Philip Chan who corrected several errors in our original understanding of their work and were generous with their time in suggesting experiments and papers to read [4]. Our appreciation does not imply that they endorse this paper.

References

[1] T. Rakthanmanon *et al.*, "Searching and mining trillions of time series subsequences under dynamic time warping," in *Proc. 18th ACM SIGKDD Intl. Conf. Knowl. Discovery and Data Mining*, Aug. 2012, pp. 262–270.

[2] A. Mueen *et al.*, "Speeding up dynamic time warping distance for sparse time series data," *Knowl. and Inf. Syst.*, vol. 54, pp. 237–263, Jan. 2018.

[3] S. Salvador and P. Chan, "FastDTW: Toward accurate dynamic time warping in linear time and space," *Intell. Data Analysis*, vol. 11, no. 5, pp. 561–580, Oct. 2007.

[4] S. Salvador and P. Chan (Mar. 2020) An Email Chain of Correspondence with Stan and Philip. [Online]. Available: https://wu.renjie.im/research/fastdtw-is-slow/emails/stan-and-philip/#philip-on-feb-27-2020-1301

[5] H. A. Dau *et al.* (Oct. 2018) The UCR Time Series Classification Archive. [Online]. Available: http://cs.ucr.edu/~eamonn/time_series_data_2018/

[6] T. Kwon, D. Jeong and J. Nam, "Audio-to-score alignment of piano music using RNN-based automatic music transcription," in *Proc. 14th Sound and Music Comput. Conf.*, Jul. 2017, pp. 380–385.