DTWNet: a Dynamic Time Warping Network

Xingyu Cai (UConn) Tingyang Xu (Tencent) Jinfeng Yi (JD.com) Junzhou Huang (Tencent) Sanguthevar Rajasekaran (UConn)

Abstract

Dynamic Time Warping (DTW) is widely used as a similarity measure in various domains. Due to its invariance against warping in the time axis, DTW provides more meaningful discrepancy measurements between two signals than other distance measures. In this paper, a learning framework based on DTW is proposed. In contrast to the previous successful usage of DTW as a loss function, we propose to apply DTW kernel as a new type of component in a neural network. The proposed framework leverages DTW to obtain a better feature extraction. For the first time, the DTW loss is theoretically analyzed, and a stochastic backpropogation scheme is proposed to improve the accuracy and efficiency of the DTW learning. We also demonstrate that the proposed framework can be used as a data analysis tool to perform data decomposition.

Introduction

In the domain of sequence data analysis, Minkowski distance, i.e. $d(x,y) = (\Sigma_{k=1}^{d} |x_k - y_k|^p)^{1/p}$, or Mahalanobis distances, i.e. $d(x,y) = ((x-y)^T\Sigma^{-1}(x-y))^{1/2}$, fail to reveal the true similarity between two targets. Dynamic Time Warping (DTW) has been proposed as an attractive alternative. DTW not only outputs the distance value, but also reveals how two sequences are aligned against each other. The standard algorithm for computing Dynamic Time Warping involves a Dynamic Programming (DP) process. With the help of $O(n^2)$ space, a cost matrix C would be built sequentially

$$C_{i,j} = ||x_i - y_j|| + \min\{C_{i-1,j}, C_{i,j-1}, C_{i-1,j-1}\} \quad (1)$$

Here $||x_i - y_j||$ denotes the norm of $(x_i - y_j)$, e.g., p-norm, $p = 1, 2$ or ∞. After performing the DP, we can trace back and identify the warping path from the cost matrix.

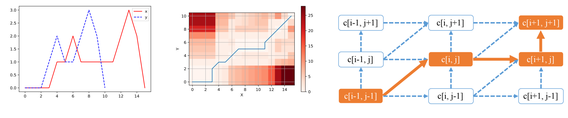

Figure 1: Illustration of DTW Computation, Dynamic Programming and Warping Path

Gradient and Backpropogation of DTWNet

One simple but important observation is that: after performing DP and obtaining the warping path, the path itself is settled down for this iteration. If the input sequences and the kernel are of lengths n and l, respectively, the length of the warping path cannot be larger than $O(n+l)$. This means that the final DTW distance could be represented using $O(n+l)$ terms, and each term is $||y_i - x_j||$ where $i, j \in \mathcal{S}$, and $\mathcal{S}$ is the set containing the indices of elements along the warping path. Since the warping path is determined, other entries in the cost matrix no longer affect the DTW distance, thus the differentiation can be done only along the path.

$$f(x,y) = ||y_0 - x_0||_2^2 + ||y_1 - x_0||_2^2 + ||y_2 - x_1||_2^2 + \ldots \Rightarrow \nabla_x f(x,y) = [2(y_0 + y_1 - 2x_0)\,,\, 2(y_2 - x_1)\,,\, \ldots]^T \quad (2)$$

Algorithm 1 DTWNet training for a classification task. Network parameters are: number of DTW kernels N_{kernel}; kernels $x_i \in \mathcal{R}^l$; linear layers with weights w.

INPUT: Dataset $Y = \{(y_i, z_i) | y_i \in \mathcal{R}^n, z_i \in \mathcal{Z} = [1, N_{\text{class}}]\}$. DTWNet $\mathcal{G}_{x,w} : \mathcal{R}^n \to \mathcal{Z}$.
OUTPUT: The trained DTWNet $\mathcal{G}_{x,w}$

Init w; For $i = 1$ to N_{kernel}: randomly init x_i; Set total # of iteration be T, stopping condition ϵ
for $t = 0$ to T **do**
  Sample a mini-batch $(y, z) \in Y$. Compute DTWNet output: $\hat{z} \leftarrow \mathcal{G}_{x,w}(y)$
  Record warping path $\mathcal{P}$ and obtain determined form $f_t(x,y)$, as in Equation 2
  Let $\mathcal{L}_t \leftarrow \mathcal{L}_{CrossEntropy}(\hat{z}, z)$. Compute $\nabla_w \mathcal{L}_t$ through regular BP.
  For $i = 1$ to N_{kernel}: compute $\nabla_{x_i}\mathcal{L}_t \leftarrow \nabla_{x_i} f_t(x_i, y)\frac{\partial \mathcal{L}_t}{\partial f_t}$ based on $\mathcal{P}$, as in Equation 2
  SGD Update: let $w \leftarrow w - \alpha\nabla_w \mathcal{L}_t$ and for $i = 1$ to N_{kernel} do $x_i \leftarrow x_i - \beta\nabla_{x_i}\mathcal{L}_t$
  If $\Delta\mathcal{L} = |\mathcal{L}_t - \mathcal{L}_{t-1}| < \epsilon$: return $\mathcal{G}_{x,w}$

Streaming DTW and Regularizer

The typical length of a DTW kernel is much shorter than the input data. Aligning the short kernel with a long input sequence, could lead to misleading results. Hence we bring the SPRING algorithm to output the patterns aligning subsequences of the original input. To prevent the kernel from learning unuseful patterns or subsequences (e.g. upsweeps that occur even in Gaussian noise) from the data, we add a regularizer in the objective function:

$$\min_{i,\Delta,x} \; (1-\alpha)\text{dtw}^2(x, y_{i:i+\Delta}) + \alpha||x_0 - x_l|| \quad (3)$$

where x is of length l and α is the hyper parameter that controls the regularizer.

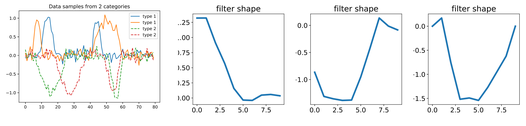

Figure 2: Effect of the streaming DTW's regularizer: from left to right, $\alpha = 0$ and 1×10^{-4} and 0.1, respectively.

Experiments and Results

1. Classification on two types of sequence data:

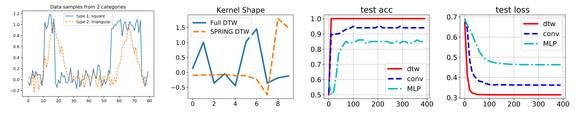

Figure 3: Performance comparison on synthetic data sequences

2. Classification on multivariate sequence data:

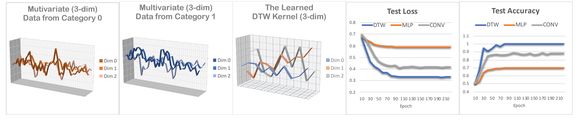

Figure 4: Multivariate DTWNet Experiment

3. Performance of the gradient computation: $\mathcal{L}_{\text{dtw}} = \frac{1}{N_{\text{class}}}\Sigma_{i=0}^{N_{\text{class}}}\frac{1}{N_i}\Sigma_{j=0}^{N_i}\text{dtw}(s_{i,j}, b_i)$.

Table 1: Barycenter Experiment Summary

	Training Set				Testing Set			
Alg	SoftDTW	SSG	DBA	**Ours**	SoftDTW	SSG	DBA	**Ours**
Win	4	23	21	**37**	11	21	22	**31**
Avg-rank	3.39	**2.14**	2.27	2.2	3.12	2.31	2.36	**2.21**
Avg-loss	27.75	26.19	26.42	**24.79**	33.08	33.84	33.62	**31.99**

DTW Loss and Convergence

Consider that for one input sequence $y \in \mathcal{R}^n$. We want to obtain a target kernel $x \in \mathcal{R}^l$ that $\min_x \text{dtw}^2(x,y)$. Without loss of generality, we assume $l \leq n$. The kernel x is randomly initialized and we perform learning through standard gradient descent. Define the DTW distance function as $d = H_y(x)$, where $d \in \mathcal{R}$ is the DTW distance evaluated by performing the Dynamic Programming operator, i.e., $d = \text{DP}(x,y)$. Since DP provides a deterministic warping path for arbitrary x, we define the space of all the functions of x representing all possible warping paths as

$$\mathcal{F}_y = \{f_y(x) | f_y(x) = \sum_{i,j} I_{ij}||(x_i - y_j)||_2^2\}$$
$$s.t.\ i \in [0, l-1];\ j \in [0, n-1];$$
$$I_{ij} \in \{0,1\};\ n \leq |I| \leq n+l;$$
$$i, j \text{ satisfy temporal order constraints.}$$

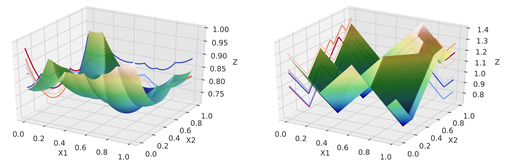

Figure 5: Piecewise quadratic or linear DTW loss

Lemma: *Warping paths number* $|\mathcal{F}_y| < 3^{n+l}$, *where* $(n+l)$ *is the largest possible path length.*
Theorem: *Assuming the starting point at coordinate* k, *i.e.* $x_k = \tilde{x}$, *is in some region* u. *Let* x *and* y *have lengths* n *and* l, *respectively, and assume that* $l < n$. *To ensure escaping from* u *to its immediate right-side neighbor region, the expected step size* $\mathbb{E}[\eta]$ *needs to satisfy:* $\mathbb{E}[\eta] > \frac{l}{2n}$.

DTW Data Decomposition

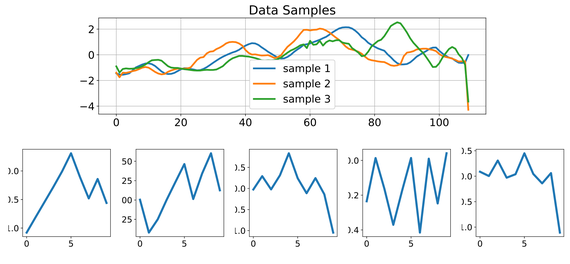

Figure 6: Illustration of DTW Decomposition