DTWNet:
a Dynamic Time Warping Network

한양대학교 산업공학과
IDSL 석사과정 고경준

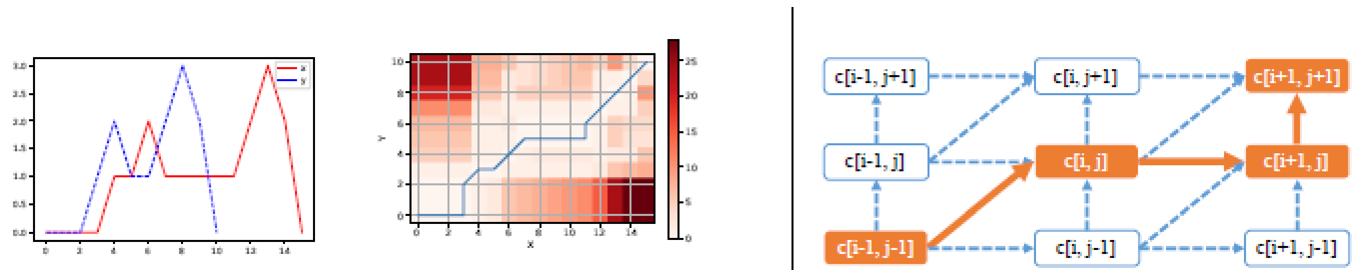
HANYANG UNIVERSITY

Contents

1. Introduction

Introduction

- 다양한 Similarity or distance metric이 존재하지만 sequence data analysis domain에서는 Doppler effect를 반영하지 못하여 부적합한 경우가 대부분임.
- (Doppler effect : shifting and scaling in the time axis)
- Dynamic Time Warping(DTW)는 signal warping의 invariance를 지님.
- DTW는 distance measure 이외에 predefined patterns을 찾아내는 feature extracting tool으로 사용되기도 함.

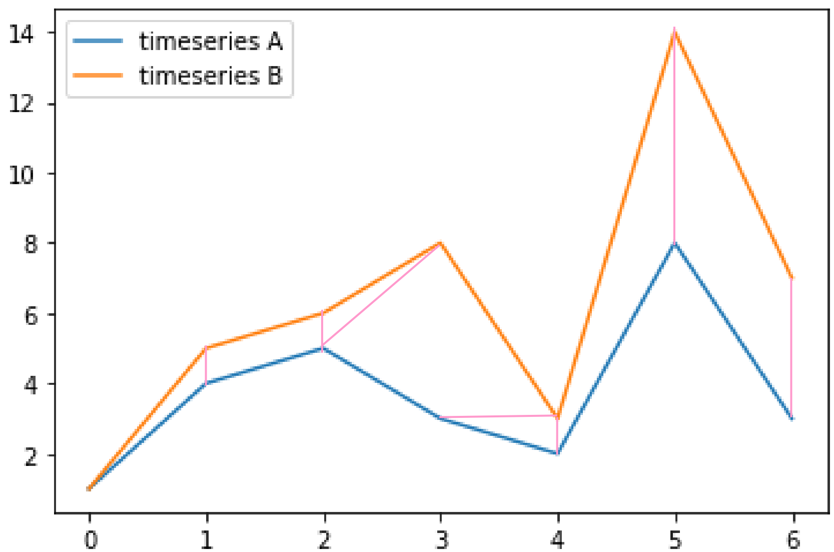


(a) DTW aligns x and y (b) DTW path of x and y (c) The path is fixed after DP

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

1. Introduction

Dynamic Programming process



Warping path

j							
7	37	13	12	16	21	13	16
14	31	10	16	16	17	12	18
3	18	8	7	5	6	11	7
8	16	7	5	7	10	7	12
6	9	3	2	4	7	8	11
5	4	1	1	3	6	9	11
1	0	3	7	9	10	17	19
	1	**4**	**5**	**3**	**2**	**8**	**3**

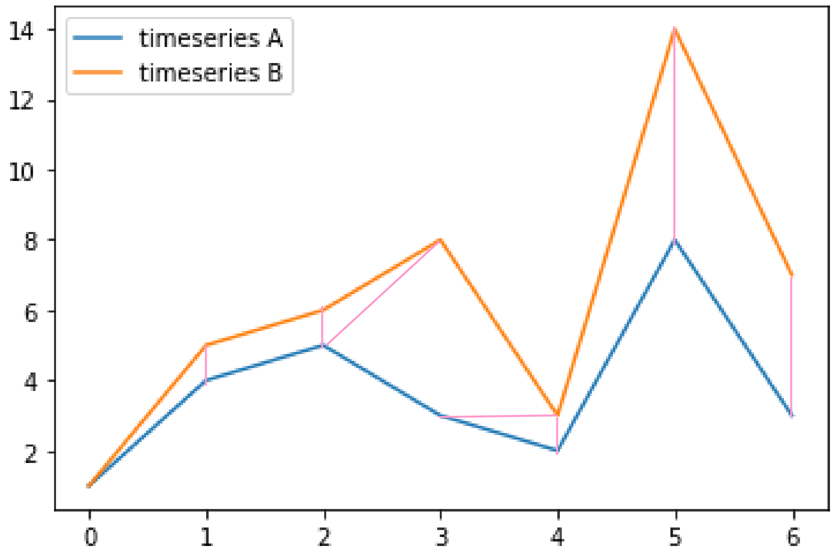
i Cost matrix

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

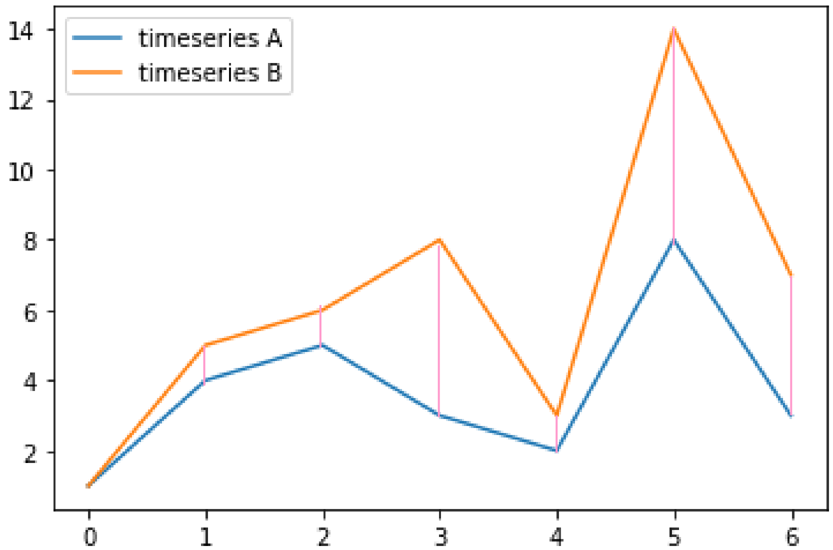
$$C_{3,2} = ||5-5|| + \min\{1, 7, 3\}$$

1. Introduction

Dynamic Programming process



DTW



Euclidean

- **Time axis에 상관없이 비슷한 형태의 데이터의 거리를 작게 측정할 수 있음.**
- **Length가 다른 데이터의 비교도 가능.**

1. Introduction

Key Contributions

- Neural network에 DTW kernels를 적용.
- Warping path를 기반으로 stochastic backpropagation method를 제안.
- 최초로 DTW loss function을 이론적으로 분석.
- Missing local features를 해결하기 위해 differentiable streaming DTW learning 제안.
- Data decomposition에 application.

2. Related Work

Introduction of Dynamic Time Warping

- Predefined patterns을 찾아내는 방식으로 feature extraction 한다.
- DP process에서 O(nl)의 time complexity 발생.
- 이미지 데이터에 관한 거리 계산이 가능한 Multi-variate DTW도 존재.

SPRING Algorithm, the Streaming Version of DTW

- Input sequence와 predefined patterns를 end-to-end로 학습하는 것이 아니라, 하나의 input에 같은 pattern이 반복되는 경우에 적용하기 위해 streaming version이 제시됨.
- 모든 subsequence를 비교할 경우 O($n^2 l$)이지만 SPRING 방식은 O(nl)

DTW as a Loss Function

- Min operation이 not continuous하므로 soft-min DTW가 적용됨.
- Soft-min idea가 사용된 사례에서 loss function을 DTW로 사용했을 때, Euclidean distance 보다 좋은 성능을 보인 사례가 존재.

3. Proposed DTW Layer and its Backpropogation

DTW Layer and Backpropagation

- Deep neural network에 DTW layer를 사용
- DTW layer는 feature를 추출하는 여러 개의 DTW kernels로 구성됨.
- DTW layer 이후에 linear layer를 추가하여 classification/regression 결과를 얻음.

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 $Y = \{(y_i, z_i) | y_i \in \mathcal{R}^n, z_i \in \mathcal{Z} = [1, N_{\text{class}}]\}$. The DTWNet dataflow can be denoted as $\mathcal{G}_{x,w} : \mathcal{R}^n \rightarrow \mathcal{Z}$.
OUTPUT: The trained DTWNet $\mathcal{G}_{x,w}$

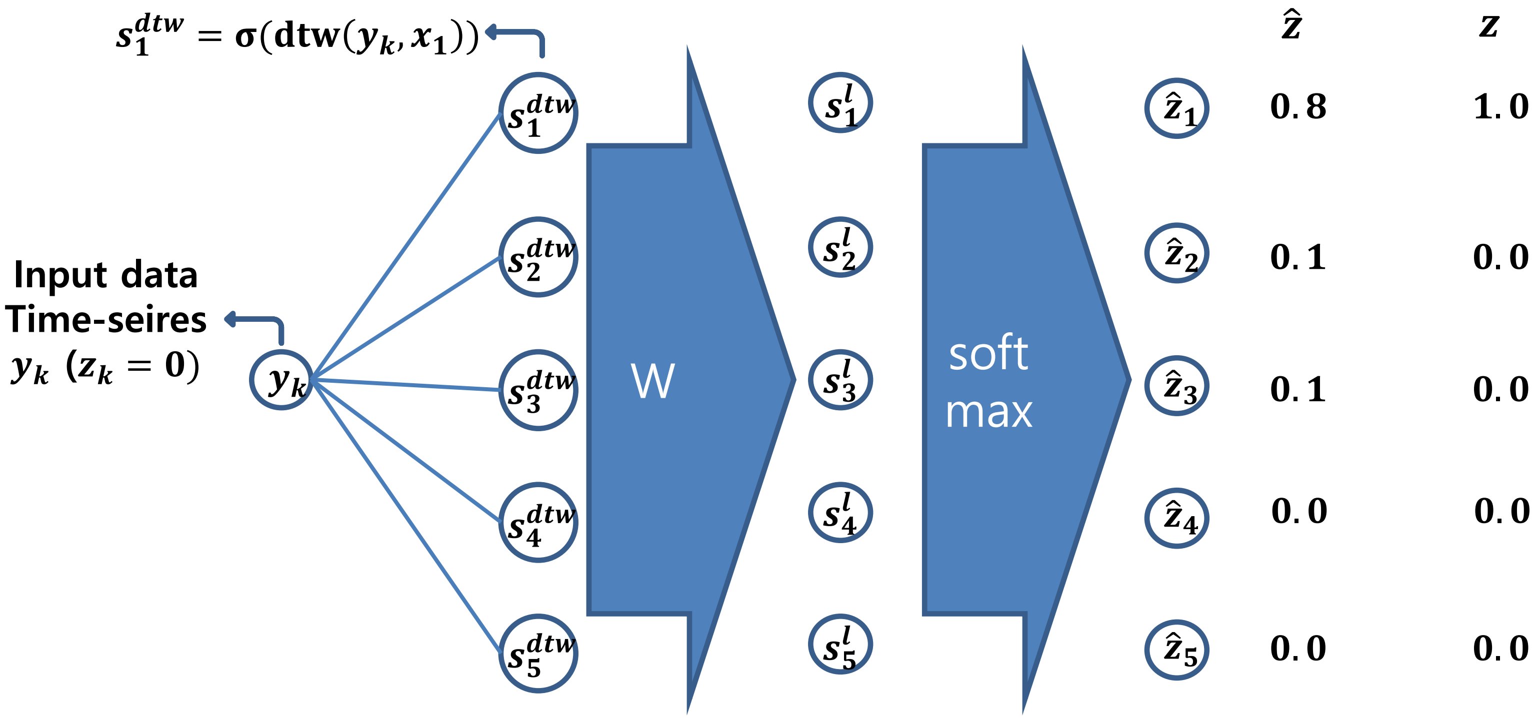
```
Init w; For i = 1 to N_kernel: randomly init x_i; Set total # of iteration be T, stopping condition ε
for t = 0 to T do
    Sample a mini-batch (y, z) ∈ Y. Compute DTWNet output: ẑ ← G_{x,w}(y)
    Record warping path P and obtain determined form f_t(x, y), as in Equation 2
    Let L_t ← L_{CrossEntropy}(ẑ, z). Compute ∇_w L_t through regular BP.
    For i = 1 to N_kernel: compute ∇_{x_i} L_t ← ∇_{x_i} f_t(x_i, y) ∂L_t/∂f_t based on P, as in Equation 3
    SGD Update: let w ← w − α∇_w L_t and for i = 1 to N_kernel do x_i ← x_i − β∇_{x_i} L_t
    If ΔL = |L_t − L_{t−1}| < ε: return G_{x,w}
```

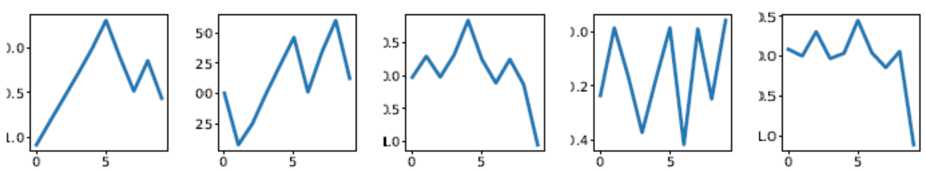
$$\text{dtw}^2(x, y) = f_t(x, y) = ||y_0 - x_0||_2^2 + ||y_1 - x_0||_2^2 + ||y_2 - x_1||_2^2 + \ldots \quad (2)$$

$$\nabla_x \text{dtw}^2(x, y) = \nabla_x f_t(x, y) = [2(y_0 + y_1 - 2x_0)\ ,\ 2(y_2 - x_1)\ ,\ \ldots]^T \quad (3)$$

3. Proposed DTW Layer and its Backpropogation

DTW Network



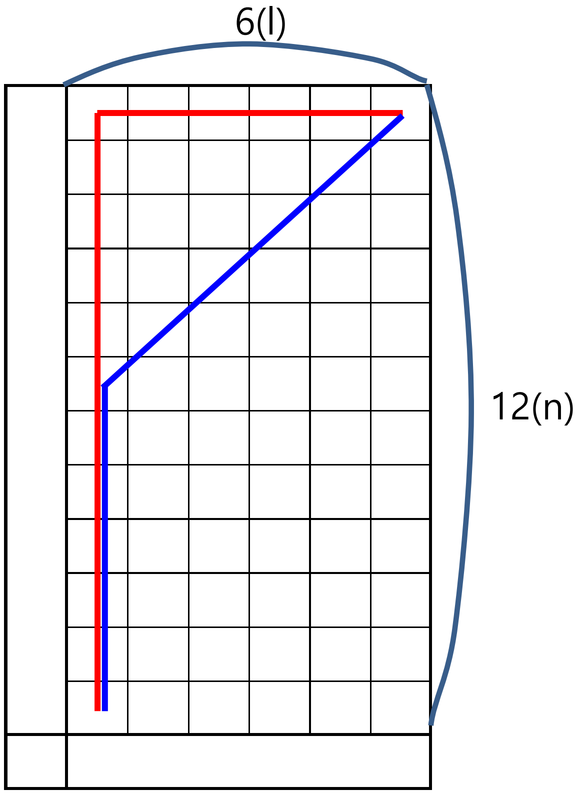


(b) Kernel 0 (c) Kernel 1 (d) Kernel 2 (e) Kernel 3 (f) Kernel 4

3. Proposed DTW Layer and its Backpropogation

Gradient Calculation and Backpropogation

- 편미분 과정은 Warping path에 관해서만 계산되기 때문에, warping path의 최대 길이인 O(n+l)time만큼 걸림.
- DP part는 not parallelizable하지만, 각 커널끼리는 parallelizable한 계산이 가능.



가장 긴 path : 6+12 = n+l
가장 짧은 path : 6 +(12-6)
= min(n,l) + (max(n,l)-min(n,l))
= max(n,l)

4. DTW Loss and Convergence

Definitions and conditions

- One Input sequence $\mathbf{y} \in \boldsymbol{R}^n$ 에 대해 $\min_{\boldsymbol{x}} \boldsymbol{dtw}^2(\boldsymbol{x}, \boldsymbol{y})$인 kernel $\mathbf{x} \in \boldsymbol{R}^l$ 를 찾는 문제. ($\boldsymbol{l} \leq \boldsymbol{n}$)

Definition 1. *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*

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

(I는 warping path에서는 1, 나머지는 0인 l행 n열 matrix.
x_i와 y_i가 mapping될 경우 x_{i+1}은 y_{i-1}와 mapping 될 수 없음.)

- DTW distance function을 $\boldsymbol{d} = \boldsymbol{H}_{\boldsymbol{y}}(\boldsymbol{x})$ 라고 정의. ($\boldsymbol{d} \in \boldsymbol{R}$)
- 따라서 임의의 x인 $\hat{x}$에 대해 $H_y(x)|_{x=\hat{x}} = f_y^{(u)}(x)|_{x=\hat{x}}$, $f_y^{(u)} \in \mathcal{F}_y$.
- 이 때, $\nabla_x f_y^{(u)}(x)|_{x=\hat{x}} = \nabla_x H_y(x)|_{x=\hat{x}}$. 을 확인.

4. DTW Loss and Convergence

Definitions and conditions

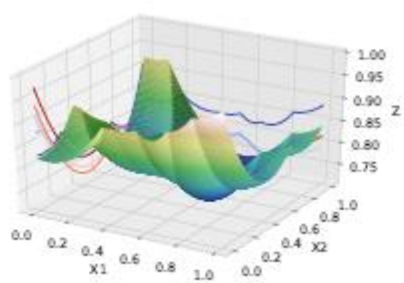
- $H_y(x)$는 x의 공간에서 not smooth하기 때문에 (4)를 만족하는 x가 존재.

$$H_y(x) = \begin{cases} f_y^{(u)}(x)|_{x=x_+} \\ f_y^{(v)}(x)|_{x=x_-} \end{cases} \quad u \neq v; \;\; f_y^{(u)}, f_y^{(v)} \in \mathcal{F}_y \tag{4}$$

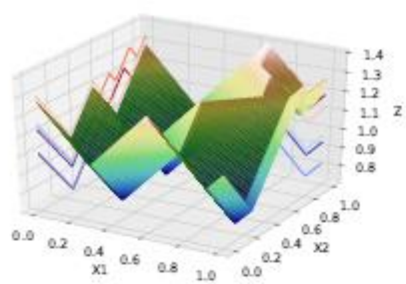
- Warping path의 계산은 3방향에서 정해지므로 Lemma 1이 성립.

Lemma 1. *Warping paths number* $|\mathcal{F}_y| < 3^{n+l}$, *where* $(n+l)$ *is the largest possible path length.*

- 따라서 $H_y(x)$는 piece-wise quadratic/linear function (2-norm/absolute DTW loss)



(a) Quadratic



(b) Linear

Figure 2

4. DTW Loss and Convergence

Escaping Local Minima

$$\mathcal{U} = \{i, j | i \neq k, j \notin [p, p+q]\}, I_{ij} \in \{0, 1\}$$

- $f_y^{(u)}$인 지역 u에서 $f_y^{(v)}$인 지역 v로 넘어가는 과정에 대한 내용.
- DP 이후 $y_{p:p+q}$가 x_k에 align되었다면, $f_y^{(u)}$를 (5)처럼 x_k 에 관해 작성 가능.

$$f_y^{(u)} = \sum_{j=p}^{p+q} (y_j - x_k)^2 + \sum_{i,j \in \mathcal{U}} I_{ij}(x_i - y_j)^2 \quad \text{and} \quad \nabla_{x_k} f_y^{(u)} = \sum_{j=p}^{p+q} 2(x_k - y_j) \tag{5}$$

- Local minimum은 $\nabla_{x_k} f_y^{(u)} = 0$ 를 만족하는 $x_k^{(u)*} = \frac{1}{q+1} \sum_{j=p}^{p+q} y_j$.

$$f_y^{(v)} = \sum_{j=p}^{p+q+1} (y_j - x_k)^2 + \sum_{i,j \in \mathcal{V}} I_{ij}(x_i - y_j)^2 \tag{6}$$

- (6)을 만족하는 local minimum을 $x_k^{(v)*}$, $\sum_{j=p}^{p+q-1} y_j$에 관한 local minimum을 $x_k^{(w)*}$ 라고 하면

$$x_k^{(u)*} = \frac{\sum_{j=p}^{p+q} y_j}{q+1} \;, \quad x_k^{(v)*} = \frac{\sum_{j=p}^{p+q+1} y_j}{q+2} \;, \quad x_k^{(w)*} = \frac{\sum_{j=p}^{p+q-1} y_j}{q} \tag{7}$$

4. DTW Loss and Convergence

Escaping Local Minima

- 지역 w, u, v가 좌에서 우로 위치한다고 가정하면 Figure 2 (c), (d)처럼 표현됨.

i.e., $x_k^1 < x_k^2 < x_k^3$, for $x_k^1 \in w, x_k^2 \in u, x_k^3 \in v$.

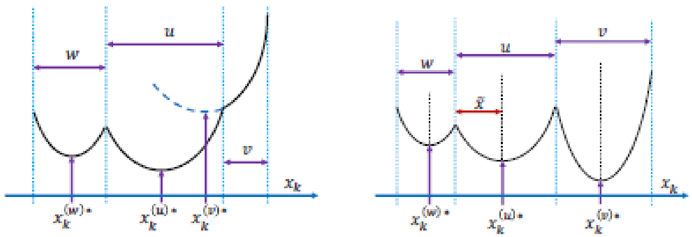


Figure 2 (c) Analysis case 1 (d) Analysis case 2

Case 1.

- Figure2c처럼 $x_k^{(v)*}$가 region v의 왼쪽에 있는 경우, 학습이 진행되면서
- region u로 돌아오기 때문에 u에서 v로 jump할 수 없음

4. DTW Loss and Convergence

Escaping Local Minima

- 지역 w, u, v가 좌에서 우로 위치한다고 가정하면 Figure 2 (c), (d)처럼 표현됨.

i.e., $x_k^1 < x_k^2 < x_k^3$, for $x_k^1 \in w, x_k^2 \in u, x_k^3 \in v$.

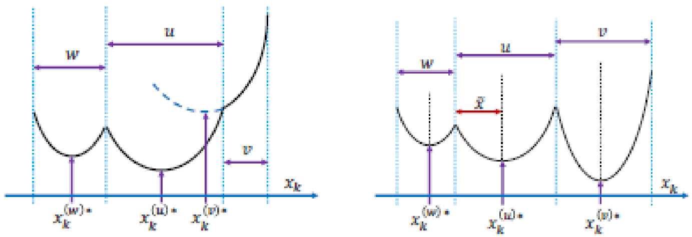


Figure 2 (c) Analysis case 1 (d) Analysis case 2

Case 2.

- Figure2d처럼 u와 v 모두 bowl 형태인 경우, 두 지역의 경계는 $x_k^{(v)*}$와 $x_k^{(u)*}$의 사이에 위치.
- 시작점 $x_k = \tilde{x} \in u$는 $x_k^{(u)*}$보다 왼쪽에 위치해야 하고 (빨간 화살표 위치)
- $x_k^{(v)*} - \tilde{x}$ 만큼 이상을 움직여야 함. (두 지역의 경계가 $x_k^{(v)*}$와 매우 인접할 수 있으므로)

4. DTW Loss and Convergence

Escaping Local Minima

- 지역 w, u, v가 좌에서 우로 위치한다고 가정하면 Figure 2 (c), (d)처럼 표현됨.

  i.e., $x_k^1 < x_k^2 < x_k^3$, for $x_k^1 \in w, x_k^2 \in u, x_k^3 \in v$.

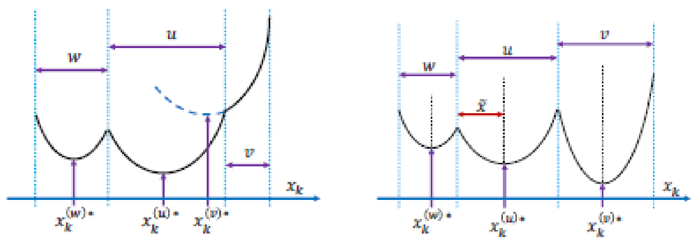


Figure 2 (c) Analysis case 1 (d) Analysis case 2

Case 3.

- v가 bowl 형태가 아니면서 $x_k^{(v)*}$가 region v의 오른쪽에 있는 경우,
- Case 2와 같은 조건 (시작 위치와 이동 거리)에서
- Figure 2c처럼 region v의 right neighbor (v+)와 합쳐진 combined region [v, v+]에 도달

4. DTW Loss and Convergence

Escaping Local Minima

- 지역 w, u, v가 좌에서 우로 위치한다고 가정하면 Figure 2 (c), (d)처럼 표현됨.

i.e., $x_k^1 < x_k^2 < x_k^3$, for $x_k^1 \in w, x_k^2 \in u, x_k^3 \in v$.

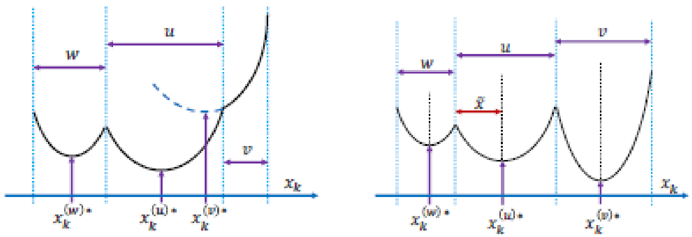


Figure 2 (c) Analysis case 1 (d) Analysis case 2

- 2,3번 case에서 u에서 v로 jump하려면 Step size가 l/2n보다 커야함.

Theorem 1. *Assume that the starting point at coordinate k, i.e. $x_k = \tilde{x}$, is in some region u where $f_y^{(u)}$ is defined in Equation $\boxed{5}$. 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: $\mathbb{E}[\eta] > \frac{l}{2n}$.*

Proof of theorem 1

A Proof of Theorem 1

Proof. As discussed above, we have two cases as follows.

Case 1: First we consider the case that w, u, v are from left to right, with their stationary points $x_k^{(w)*} < x_k^{(u)*} < x_k^{(v)*}$. A standard gradient descent update is $\tilde{x} \leftarrow \tilde{x} - \eta \nabla_{x_k} f_y^{(u)}|_{x_k=\tilde{x}}$. To ensure one step update could make $\tilde{x}$ jump from u to v, we obtain:

$$
\begin{aligned}
&\tilde{x} - \eta \nabla_{x_k} f_y^{(u)}|_{x_k=\tilde{x}} \geq x_k^{(v)*} \\
\Rightarrow\ &\tilde{x} - \eta \sum_{j=p}^{p+q} 2(\tilde{x} - y_j) \geq x_k^{(v)*} \quad \text{(use Equation 5)} \\
\Rightarrow\ &(1 - 2\eta(q+1))\tilde{x} + 2\eta(q+1)\frac{1}{q+1}\sum_{j=p}^{p+q} y_j \geq x_k^{(v)*} \\
\Rightarrow\ &(1 - 2\eta(q+1))\tilde{x} \geq x_k^{(v)*} - 2\eta(q+1)x_k^{(u)*} \quad \text{(use Equation 7)}
\end{aligned}
\tag{11}
$$

4. DTW Loss and Convergence

Proof of theorem 1

Now we have two possibilities:

Possibility (a): $1 - 2\eta(q+1) > 0$**:**

$$\text{Inequality 11} \Rightarrow \tilde{x} \geq \frac{x_k^{(v)*} - 2\eta(q+1)x_k^{(u)*}}{1 - 2\eta(q+1)}$$

To guarantee the gradient direction points to neighbor v, the starting point $\tilde{x}$ has to be to the left of $x_k^{(u)*}$, thus:

$$\begin{aligned} &\frac{x_k^{(v)*} - 2\eta(q+1)x_k^{(u)*}}{1 - 2\eta(q+1)} \leq \tilde{x} < x_k^{(u)*} \\ &\Rightarrow x_k^{(v)*} - 2\eta(q+1)x_k^{(u)*} < (1 - 2\eta(q+1))x_k^{(u)*} \\ &\Rightarrow x_k^{(v)*} < x_k^{(u)*} \end{aligned} \tag{12}$$

This is contradictory to the assumption that $x_k^{(w)*} < x_k^{(u)*} < x_k^{(v)*}$, and thus is not valid.

4. DTW Loss and Convergence

Proof of theorem 1

Possibility (b): $1 - 2\eta(q+1) < 0$**:**

$$\text{Inequality 11} \Rightarrow \ \tilde{x} \leq \frac{x_k^{(v)*} - 2\eta(q+1)x_k^{(u)*}}{1 - 2\eta(q+1)}$$

Due to the fact that $\tilde{x}$ is inside region u, which must be somewhere to the right of $x_k^{(w)*}$ (with the assumption that w has a bowl-shape), we have:

$$\begin{aligned}
&\frac{x_k^{(v)*} - 2\eta(q+1)x_k^{(u)*}}{1 - 2\eta(q+1)} \geq \tilde{x} > x_k^{(w)*} \\
&\Rightarrow x_k^{(v)*} - 2\eta(q+1)x_k^{(u)*} < (1 - 2\eta(q+1))x_k^{(w)*} \\
&\Rightarrow x_k^{(v)*} - x_k^{(w)*} < 2\eta(q+1)(x_k^{(u)*} - x_k^{(w)*}) \\
&\Rightarrow \eta > \frac{1}{2(q+1)}\left(\frac{x_k^{(v)*} - x_k^{(w)*}}{x_k^{(u)*} - x_k^{(w)*}}\right)
\end{aligned} \tag{13}$$

Recall that q is an integer and $q \geq 0$, thus

$$1 - 2\eta(q+1) < 0 \ \Rightarrow \ \eta > \frac{1}{2(q+1)} \tag{14}$$

Also notice that

$$x_k^{(w)*} < x_k^{(u)*} < x_k^{(v)*} \ \Rightarrow \ \frac{x_k^{(v)*} - x_k^{(w)*}}{x_k^{(u)*} - x_k^{(w)*}} > 1 \tag{15}$$

IVERSITY

4. DTW Loss and Convergence

Proof of theorem 1

Case 2: here w, u, v are from right to left, and $x_k^{(w)*} > x_k^{(u)*} > x_k^{(v)*}$. This is very similar to Case 1, so we omit the details and provide the final result as

$$\eta > \frac{1}{2(q+1)}\left(\frac{x_k^{(w)*} - x_k^{(v)*}}{x_k^{(u)*} - x_k^{(v)*}}\right) \quad \text{and} \quad \frac{x_k^{(w)*} - x_k^{(v)*}}{x_k^{(u)*} - x_k^{(v)*}} > 1 \tag{16}$$

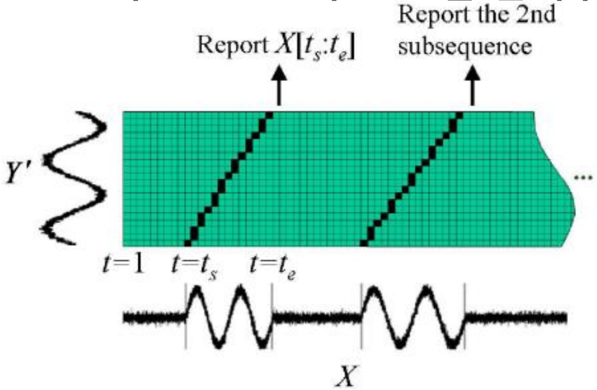
We will arrive at the same result: $\eta > \frac{1}{2(q+1)}$.

Note that the length of the pattern x is l and the length of input y is n. As a result, the expected number of elements in y aligned to a single x_i, $i \in [0, l-1]$ should be n/l, i.e. $\mathbb{E}[q] = n/l - 1$. Taking expectation on both sides of the above inequality, we obtain $\mathbb{E}[\eta] > \frac{1}{2(n/l-1+1)} = \frac{l}{2n}$. $\square$

5. Streaming DTW Learning

Spring algorithm

- Input data에 pattern이 여러 번 반복되는 경우엔 kernel이 전체 input에 대한 pattern을 학습하려고 하기 때문에 문제가 발생.
- Window를 정해서 subsequence와 kernel간의 DTW를 줄여주는 방식.



$$x^* = \arg\min_{i,\Delta,x} \operatorname{dtw}^2(x, y_{i:i+\Delta})$$

- Network를 통해 DTW distance를 계산하여 지금까지 중 가장 작은 경우(best i, Δ)를 기억.
- 다음 input이 들어오면 i, Δ를 다르게 설정하여 학습한 뒤, output distance가 minimum distance보다 작으면 해당 i, Δ를 (best i, Δ)로 기억 후 kernel 학습

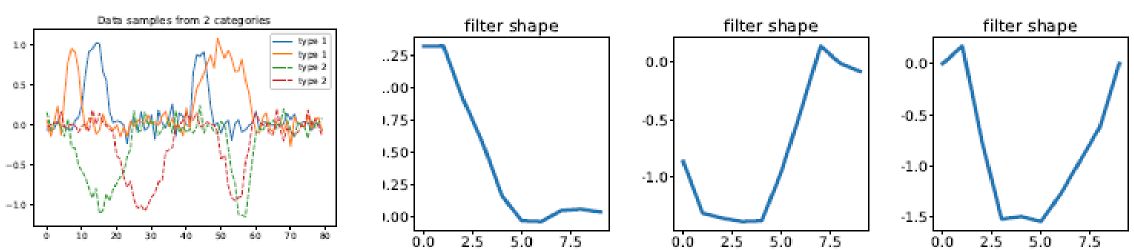
5. Streaming DTW Learning

Regularizer in Streaming DTW

- SPRING은 input sequence 중 반복되는 pattern을 찾아내는 알고리즘이기 때문에 일반적으로 나타날 수 있는 오르거나 내리는 형태도 pattern으로 인식하게 됨.
- 또는 pattern의 오르거나 내리는 일부분만 학습하게 됨.

$$\min_{i,\Delta,x} \ (1-\alpha)\mathrm{dtw}^2(x, y_{i:i+\Delta}) + \alpha||x_0 - x_l|| \qquad (9)$$

- Kernel의 처음과 끝의 거리를 줄여주어 일부분만 학습하는 오류를 방지.



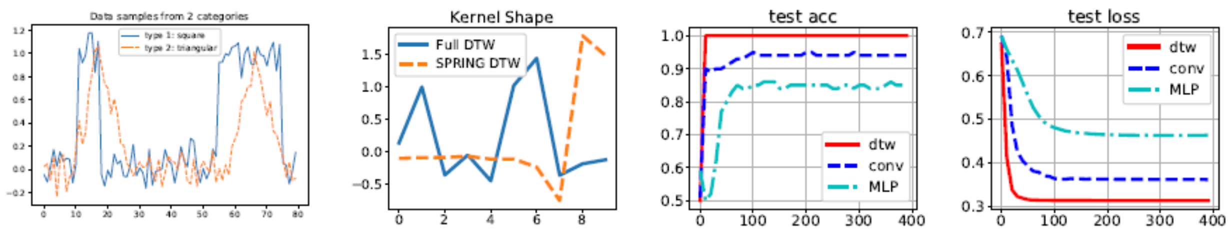
(a) Data samples (b) No reg (c) Week reg (d) Strong reg

- 또한 pattern에 대한 사전 정보가 있는 경우, 그와 관련된 regularizer를 추가할 수 있음.

6. Experiments and Applications

Comparison with Convolution Kernel

- 처음 설명한 End-to-end 방식을 Full DTW, streaming 방식을 SPRING DTW라고 표현.
- 인공적인 timeseries input data를 생성하여 성능을 비교.
- 각 time series는 Pattern 1(half square), Pattern 2(upper triangle) 중 1개가 2번 발생.
- 발생하는 Pattern의 위치와 길이는 random하고, Gaussian noise가 추가된 형태.



(a) Data samples (b) Learned kernels (c) Test acc (d) Test loss

Figure 4: Performance comparison on synthetic data sequences (400 iterations)

- Training set : 각 pattern 별 50개씩 총 100개
- Test set : 각 pattern 별 50개씩 총 100개
- Full DTW, SPRING DTW, convolution kernel (length 10)를 각각 학습시킴.
- Set α = 0.1 in SPRING DTW, 3 linear layers

6. Experiments and Applications

Evaluation of Gradient Calculation

- Proposed BP schem의 accurac를 비교하기 위한 실험.
- 이를 위해 여러가지 방식으로 barycenter를 계산하고 최종 loss를 비교해 봄.

$$\mathcal{L}_{\text{dtw}} = \frac{1}{N_{\text{class}}} \sum_{i=0}^{N_{\text{class}}} \frac{1}{N_i} \sum_{j=0}^{N_i} \text{dtw}(s_{i,j}, b_i) \quad (10)$$

(i class를 가지는 j번째 input sequence와 barycenter간의 dtw
N_{class}는 category 개수, N_i는 i class인 sequence의 개수)

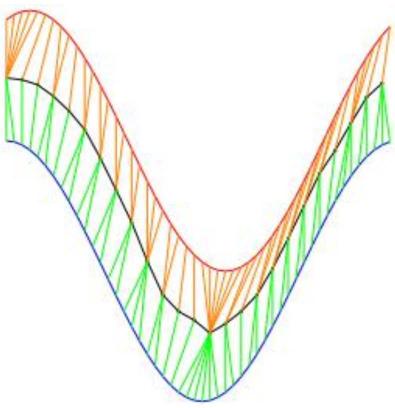


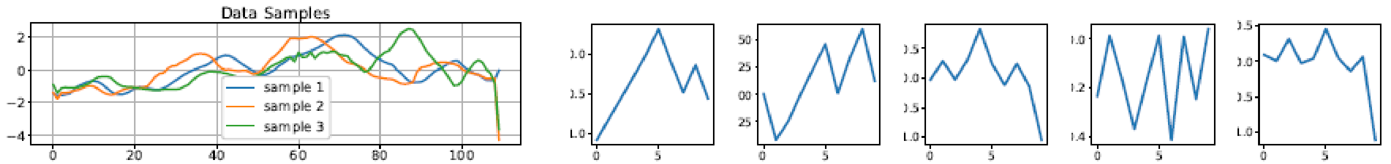
Table 1: Barycenter Experiment Summary

	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**

6. Experiments and Applications

Application of DTW Decomposition

- DTW layer를 여러 겹 쌓아서 decomposition effect를 얻을 수 있음.
- 앞부분의 DTW layer residual을 다음 DTW layer에 forwarding하는 idea.



(a) Samples of Haptics dataset (b) Kernel 0 (c) Kernel 1 (d) Kernel 2 (e) Kernel 3 (f) Kernel 4

Figure 5: Illustration of DTW Decomposition

- 5개의 DTW layer에 1개의 kernel만 배정하여 학습하면, 일반적으로
- first layer에서는 전체적인 pattern (low-frequency)을,
- last layer에서는 high-frequency pattern을 학습하는 모습을 보임.

감사합니다.