공학석사학위논문

동적 시간 정합 기반 손실함수의 개선을 통한 딥러닝 모델의 시계열 예측 성능 제고

Improving Time Series Forecasting Performance of Deep Learning Models by Enhancing Dynamic Time Warping based Loss Function

2023 년 2 월

서울대학교 대학원

산업공학과

김 재 희

동적 시간 정합 기반 손실함수의 개선을 통한 딥러닝 모델의 시계열 예측 성능 제고

Improving Time Series Forecasting Performance of Deep Learning Models by Enhancing Dynamic Time Warping based Loss Function

지도교수 이 경 식

이 논문을 공학석사 학위논문으로 제출함

2022 년 12 월

서울대학교 대학원

산업공학과

김 재 희

김재희의 공학석사 학위논문을 인준함

2023 년 1 월

위 원 장	조 성 준	(인)
부위원장	이 경 식	(인)
위 원	박 종 헌	(인)

초록

다시점 시계열 예측은 시간에 따라 변화하는 기록 데이터를 획득할 수 있는 다양한 산업 및 연구 분야에서 매우 활발히 연구되어 오던 주제이다. 현재는 딥러닝 모델을 이용해 시계열의 고유한 특성인 주기성, 추세성, 비규칙성 등의 시간적 역학(temporal dynamics)을 학습하는 연구 방법론이 일반적이나, 모델의 예측 결과를 평가하고 학습 방법 및 방향을 결정하는 손실함수에 대한 연구는 아직까지 많지 않다.

본 논문에서는 시계열 예측 용 딥러닝 모델의 학습을 위한 개선된 동적 시간 정합(Dynamic Time Warping, DTW) 기반 손실함수를 제시한다. 미분 가능한 동적 시간 정합에 거리 기반 가중 방법(Weighted DTW)과 주변 시점들을 함께 고려하는 모양 기술자를 사용하는 방법(Shape DTW)를 적용하여 목표 시계열의 모양(변화의 크기와 시점)을 더욱 정확하게 예측할 수 있도록 의도하였다. 제시한 손실함수를 여러 딥러닝 모델과 서로 다른 특징을 갖는 실제 데이터 셋들에 적용하고 유클리드 거리 기반 손실함수 및 기존 동적 시간 정합 기반 손실함수와의 비교를 통해 예측 성능이 향상됨을 보였다. 또한 정량적 관점에서는 다양한 평가지표를 사용하여 시계열 예측을 여러 관점에서 평가하였으며, 정성적 평가를 통해 제시한 손실함수가 시계열의 급격한 변화를 더욱 잘 예측한다는 것을 보여 기존 손실함수를 충분히 대체할 수 있음을 확인하였다.

주요어: 다시점 시계열 예측, 동적 시간 정합, 딥러닝 모델, 손실 함수, 산업공학

학번: 2021-27604

목차

표 목차

그림 목차

제 1 장 서론

시계열 데이터는 일정한 시간 간격으로 수집되고 관찰된 데이터들이 시간순으로 정렬 되어 표현된 데이터이다 [52]. 그리고 이 시계열 데이터를 분석하여 미래 어느 시점까지의 목표 변수의 값들 또는 분포들을 예측하는 것은 다양한 산업 분야에서 깊은 관심을 가지는 연구 분야이다[69]. 특히 생물학, 의학, 기상학 뿐 아니라 소매업과 금융업 등의 분야에서 시계열 예측 연구는 그 유서가 깊다 [20]. 이 때, 시계열 예측 결과의 정확도는 의사 결정의 질과 직결된다 [33]. 예를 들어 금융 분야에서는 주식의 가격, 암호화폐 가격, 금 가격, 원·달러 환율 등을 주로 예측하는 것을 목표로 하는데, 예측치의 정확도가 높아지는 것은 곧 투자의 이익이 높아지는 것 혹은 투자의 리스크가 낮아지는 것을 의미한다. 이러한 면에서 시계열 데이터가 딥러닝 분야 연구에서 주로 활용되는 텍스트 데이터나 이미지 데이터보다 사회, 경제적 잠재가치가 크다고 평가 되기도 한다 [13]. 본 연구는 궁극적으로는 시계열 예측 성능을 제고하여 산업적 가치로 환원하는 것을 목적으로 하며 특히 예측하기 어려운 시계열 데이터의 예측 성능을 개선하고자 한다. 1.1 절에서 연구의 배경과 동기에 대해 소개한다.

1.1 연구 배경 및 동기

많은 연구들이 단순히 주어진 데이터에서의 성능을 높이는 것에 주로 집중하고 각각의 시계열이 지니는 다양한 특성을 분석하는 것에는 소홀히 해왔다 [52]. 달리 말하면, 각각의 시계열 데이터의 특성을 면밀히 분석하고 비교하지 않은 상태에서는, 대표적인 몇 개 유형의 시계열 데이터에서 좋은 예측 성능을 보인 모델이 특별한 성격을 지닌 새로운 시계열 데이터에서도 높은 예측 성능을 보일 것이란 보장을 하기 어렵다. 실제로 시계열 데이터의 대표적인 특성인 추세, 계절성, 비규칙성 중 계절성과 추세가 분명하지 않은 금융 분야 시계열 데이터 예측의 난이도는 급격하게 상승한다 [49]. 딥러닝 모델에 대한 연구 이외에도 모델이 학습하는데에 영향을 주는 손실함수에 대한 연구가 이러한 문제 상황의 해답이 될 수 있다 [16].

따라서 본 연구에서는 다양한 유형의 시계열 데이터의 특성을 비교 분석한 후, 딥러닝 모델이 변동성이 더 큰 시계열 데이터를 더욱 잘 예측할 수 있도록 하는 새로운 손실함수를 제시하고자 한다. 비교적 짧은 중단기 예측 상황에서 제안한 손실함수를 다양한 딥러닝 모델에 적용한 후 기존에 주로 사용되어 왔던 유클리드 거리 기반 손실함수를 적용했을 때와 비교하여 실제로 변동성이 큰(급격한 변화를 많이 보이는) 시계열 데이터에서 높은 예측 성능을 보이는지 확인하고자 한다.

시계열 예측 시, 딥러닝 모델을 학습시킬 때는 일반적으로 유클리드 거리 기반 손실함수가 주로 쓰인다. 그런데 가장 대표적인 유클리드 거리기반 손실함수인 MSE를 이용해 딥러닝 모델을 학습 시킨 경우 단순히 해당 시점에서의 오차의 크기에만 집중하기 때문에 예측이 평활되는 경향이 있고 결과적으로 실제 예측해야하는 시계열의 모양을 충분히 반영하지 못한다는 한계가 존재한다 [16, 31]. 따라서 단순히 해당 시점에서의 오차의 크기를 고려하는 것이 아닌 시계열 곡선 모양 자체의 유사도 혹은 차이를 반영할 수 있는 손실함수를 사용하는 것을 고려해볼 필요가 있다. 또한 예측된

결과를 평가할 때에도, 시계열의 특성이나 사용하려는 목적에 따라 유클리드 거리 기반 함수보다 Dynamic Time Warping 같은 시계열의 유사도를 측정하는 새로운 지표가 유리할 가능성도 검토해봐야 한다. 때에 따라서는 각 시점에서의 절대적 오차만을 확인하는 것이 시계열 예측의 질을 충분히 대표하지 못할 수 있기 때문이다 [31]. 위의 문제 인식에 따라, 본 연구는 Dynamic Time Warping 에 기반한 손실함수를 개선하여 다양한 유형의 시계열 데이터에서 딥러닝 모델을 학습시키고 특정 성질을 지니는 시계열에서 본 연구가 제시한 손실함수가 더 높은 예측 성능을 보이는지 확인하고자 한다.

1.2 연구 목적

본 연구에서는 딥러닝 모델이 시계열의 급격한 변화를 더욱 잘 예측 할 수 있도록 하는 개선된 손실함수를 제안한다. 시계열 데이터를 입력으로 받을 수 있는 딥러닝 모델과 개선된 동적 시간 정합 기반 손실함수를 이용하여 시계열의 모양과 변화 시점을 더욱 정확하게 예측 가능한 방법을 연구한다. 연구의 실제적 적용을 위하여 서로 다른 특징을 갖는 4개의 시계열 데이터 셋에서 실험을 진행하며, 모델에 관계 없이 적용할 수 있고 좋은 성능을 가짐을 보이기 위해 모델 3개의 딥러닝 모델 각각에 적용해 본다. 기존 방법들과의 결과 비교를 통해 예측 성능이 향상됨을 보이고 시계열 예측을 위한 딥러닝 모델의 손실함수 연구의 타당성을 보이는 것을 목적으로 한다. 정리하면 다음과 같다.

(a) 시계열 예측용 딥러닝 모델의 손실함수를 개선한다.

(b) 다양한 특징을 갖는 데이터 셋의 충분한 이해와 분석 이후 제안 기법을 적용한다.

(c) 제안한 기법의 효과를 입증하기 위해 다양한 특징을 갖는 데이터 셋들에 대한 비교 실험을 진행한다.

1.3 문제 정의

본 연구에서 풀고자 하는 문제는 단변량 시계열 데이터를 이용한 다시점 예측(Multi-step forecasting) 문제로 정의한다. 시계열 예측 모델은 N 시점을 입력으로 하여 그 다음 τ 시점을 예측한다. 본 실험에서 N 은 $\{24, 48, 72\}$ 의 원소 중 중 가장 좋은 성능을 내는 값으로 선택하고, $\tau = 24$ 로 정한다. 이를 수식으로 나타내면 수식 1.1 과 같다. 여기서 t는 현재 시점, F는 단변량 시계열 $y \in \mathbb{R}^{N\times 1}$를 입력으로 하여 $\hat{y} \in \mathbb{R}^{\tau\times 1}$ 를 예측하는 예측 모델을 타나낸다.

$$\hat{y}_{t+1}, ..., \hat{y}_{t+\tau} = F(y_{t-N+1}, ..., y_t) \tag{1.1}$$

이 문제를 묘사한 그림이 그림 1.1에 제시되어 있다.

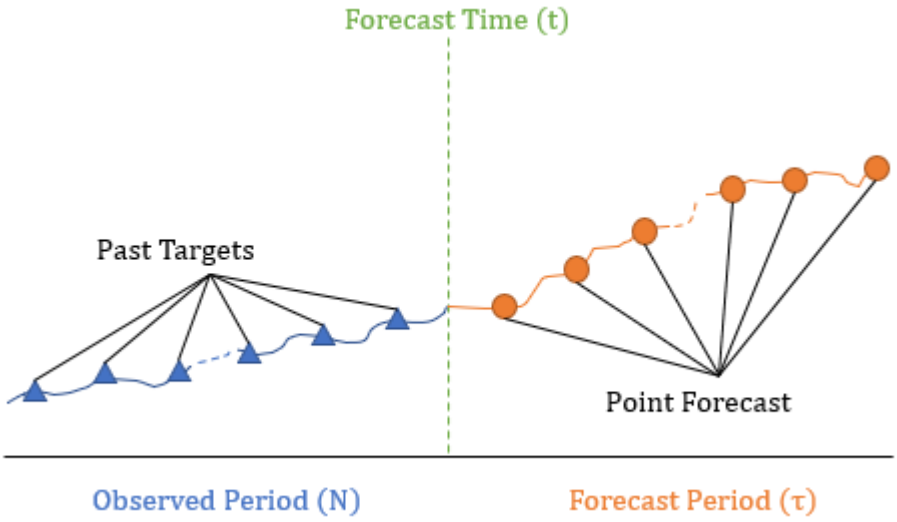


Figure 1.1: 다시점 시계열 예측 그림

1.4 논문구성

본 논문은 다음의 총 5 장으로 구성된다. 제 2장에서는 본 연구에서 활용하고자 하는 동적 시간 정합(Dynamic Time Warping)과 시계열 예측을 위한 딥러닝 모델에 대해 소개하고, 관련된 선행 연구를 살펴본다. 제 3장에서는 동적 시간 정합 기반 손실함수를 개선한 제안 기법들의 원리와 특징에 대해 자세히 설명한다. 제 4장에서는 연구에 사용된 4개의 데이터 셋과 3개의 딥러닝 모델 및 실험 세팅에 대해 설명한 후 실험을 통해 제안 기법을 기존 기법과 비교해 본다. 마지막으로 제 5장에서는 연구의 결론과 한계점에 대해 논의하고 향후 연구방향을 제시한다.

제 2 장 배경 이론 및 관련 연구

2.1 배경 이론

2.1.1 동적 시간 정합 (Dynamic Time Wariping)

동적 시간 정합[7] 은 매우 효율적인 시계열 유사도 측정 알고리즘으로 1960년대에 처음 등장 [5]하여 70년대에 음성인식 분야에서 큰 각광을 받았고 [41] 지금까지 많은 분야에서 널리 사용되고 있다 [39]. 주요 활용 분야로는 수화 인식, 제스처 인식, 수기와 온라인 서명 매칭, 시계열 클러스터링을 통한 데이터 베이스 검색, 컴퓨터 비전과 영상, 화학공학에서의 단백질 서열 정렬, 음악과 신호 처리 등이 있다 [50]. 동적 시간 정합은 다른 시점 간 유사한 모양을 탐지하기 위해 시계열 시간 축의 유연한 변환을 허용함으로써 시간 축에서의 왜곡 및 밀림의 효과를 최소화하는 알고리즘이다 [28]. 예를 들어, 두 시계열 간의 유사도를 계산할 때, 유클리드 거리를 사용하게 된다면, 같은 시점들에 대한 거리를 계산하게 된다. 그런데 시점 이동이 생기거나 시간 축에서의 확대 및 축소, 즉 속도의 변화가 생길 때는 시계열의 유사성을 제대로 판단할 수 없다는 단점이 있다. 이에 반해 동적 시간 정합은 같은 시간대 뿐 아니라 주변 시점의 값까지 거리를 비교한 후, 더 거리가 짧은 요소와의 매칭을 통해 시간축 왜곡의 영향을 최소화하여 위의 문제를 해결한다. 이 덕분에 동적 시간 정합을 이용하면 유클리드 거리와 달리 길이가 다른 시계열 간의 유사도를 측정할 수 있다는 특장점을 가진다. 유클리드 거리와 동적 시간 정합 거리를 비교한 개념이 그림2.1에 제시되어 있다.

Figure 2.1: 유클리드 거리와 동적 시간 정합 거리의 비교 그림 [28]

동적 시간 정합은 다중 스케일링 [40, 48] 과 동적 프로그래밍 기법으로 $O(NM)$ 의 시간 복잡도에 최적해를 계산할 수 있음이 증명되었으며, 다음으로 동적 시간 정합 거리를 구하는 방법을 알아본다.

두 시계열 $X = (x_1, x_2, ..., x_N)$, $N \in \mathbb{N}$ 과 $Y = (y_1, y_2, ..., y_M)$, $M \in \mathbb{N}$ 가 있다고 할 때, X와 Y 각각의 모든 원소간의 거리(L_2-norm)를 계산한 거리 행렬(distance matrix) $C \in \mathbb{R}^{N \times M}$ 를 만든다. 이 거리 행렬은 X와 Y 두 시계열의 정렬을 위한 지역 비용 행렬(local cost matrix)로 명명한다. $c_{i,j}$ 는 각 원소 x_i, y_j 들의 L_2-norm을 의미한다.

$$C \in \mathbb{R}^{N \times M} : \ c_{i,j} = \|x_i - y_j\| \, , \ i \in [1 : N], \ j \in [1 : M] \tag{2.1}$$

지역 비용 행렬이 준비되면, 알고리즘은 전체 비용이 최소가 되도록 하는 워핑 경로(warping path)를 찾게 된다. 지역 비용 행렬의 히트맵과 히트맵 상에서 가장 비용이 작게 되는 정렬 경로(최적 워핑 경로)를 나타낸 그림은 그림 2.2 과 같다.

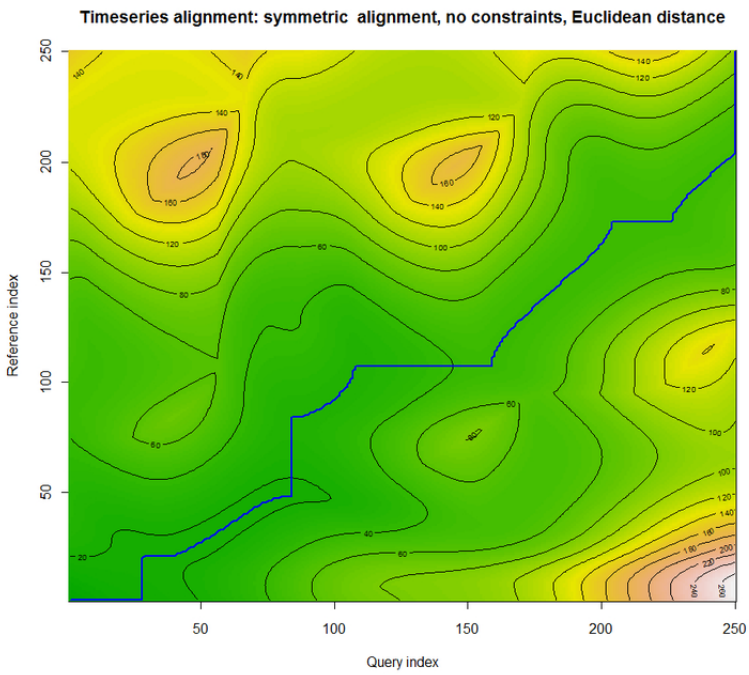


Figure 2.2: 지역 비용 행렬의 히트맵 [50]

다르게 표현하면, 동적 시간 정합 알고리즘이 산출하는 경로는 수식 2.2에서 보듯 점들의 나열인 W이고 W의 모든 점 $w_i, i \in [1, k]$ 와 w_i 의 x좌표, y좌표를 나타내는 n_i, m_i에 대하여 다음과 같은 3개의 조건을 만족해야 한다.

(a) 경계 조건 (Boundary condition) : $w_1 = (1, 1), w_K = (N, M)$. 시작점과 끝점에 대한 조건이다.

(b) 단조성 조건 (Monotonicity condition) : $n_1 \leq n_2 \leq ... \leq n_k$, $m_1 \leq m_2 \leq ... \leq m_k$. 워핑 경로는 음의 방향으로 연결되지 않는다.

(c) 연속성 조건 (Continuity condition) : $w_{l+1} - w_l \in (1, 1), (1, 0), (0, 1)$. 워핑

경로는 인접한 경로로 제한된다.

$$W = (w_1, w_2, ..., w_k),\ w_l = (n_l, m_l) \in [1:N] \times [1:M],\ l \in [1:K] \tag{2.2}$$

이 워핑 경로는 위의 세 가지 조건을 만족하는 가능한 모든 연결된 이진 경로 행렬의 집합은 $\mathcal{A}_{N,M} \subset \{0,1\}^{N \times M}$ 로 나타낼 수 있다. 이 워핑 경로의 비용함수는 경로 행렬과 지역 비용 행렬의 내적인 $\langle A, C \rangle$ 로 나타낼 수 있다.

X, Y 두 시계열에서 가능한 모든 경로 중 가장 비용이 적은 경로가 최적 경로가 되고 최적 경로 행렬을 A^* 라 한다. 그런데 가능한 모든 경로의 집합 $\mathcal{A}_{N,M}$ 의 집합의 크기 (cardinality)는 델라노이 수[4]로 표현 시, $delannoy(N-1, M-1)$ 로 알려져 있고 모든 경로의 비용을 계산하는 것은 지수 시간 복잡도를 가지기 때문에 계산량을 줄이기 위해 동적 프로그래밍 (Dynamic Programming, DP) 기반 알고리즘을 사용하여 $O(NM)$ 의 시간 복잡도로 최적 경로를 구할 수 있다. 동적 시간 정합 거리 함수는 수식 3.1 와 같다.

$$DTW(X, Y) := \langle A^*, C \rangle = min_{A \in \mathcal{A}_{N,M}}\ \langle A, C \rangle \tag{2.3}$$

동적 프로그래밍으로 계산하기 위한 전역 비용 행렬(Global cost matrix) D는 다음의 수식 3.2으로 구한다.

$$
\begin{gathered}
D(1,j) = \sum_{k=1}^{j} c(x_1, y_k), j \in [1, M] \\
D(i,1) = \sum_{k=1}^{i} c(x_k, y_1), i \in [1, N] \\
D(i,j) = min\left\{D(i-1,j-1), D(i-1,j), D(i,j-1)\right\} + c(x_i, y_j) \\
,\ i \in [2, N],\ j \in [2, M]
\end{gathered}
\tag{2.4}
$$

전역 비용 행렬 D를 구하는 pseudo-code는 아래와 같다.

Algorithm 1 전역 비용 행렬 (X, Y, C)

```
n ← |X|
m ← |Y|
dtw[ ] ← new[N × M]
dtw(0,0) ← 0
for i = 1; i ≤ N; i++ do
    dtw(i,1) ← dtw(i−1,1) + c(i,1);
end for
for j = 1; j ≤ M; j++ do
    dtw(1,j) ← dtw(1,j−1) + c(1,j);
end for
for i = 1; i ≤ N; i++ do
    for j = 1; j ≤ M; j++ do
        dtw(i,j) = min{dtw(i−1,j−1), dtw(i−1,j), dtw(i,j−1)} + c(i,j);
    end for
end for
return dtw
```

위 알고리즘의 결과, 예시 그림 2.3 와 같이 동적 시간 정합을 완료할 수 있다. 최적 경로는 전역 비용 행렬을 백트래킹 기법으로 $w_{end} = (M, N)$ 으로부터 시작해 $w_{start} = (1, 1)$ 까지를 추적하며 순차적으로 구할 수 있다. 이러한 w들은 최적 경로를 이루며 이를 구하는 알고리즘은 아래 슈도 코드로 설명한다.

Algorithm 2 최적 워핑 경로 (dtw)

```
path[ ] ← new array,
i = rows(dtw)
j = columns(dtw)
while (i > 1) & (j > 1) do
    if i == 1 then
        j = j − 1
    else if j == 1 then
        i = i − 1
    else
        mindtw = min {dtw(i − 1, j − 1), dtw(i − 1, j), dtw(i, j − 1)}
        if dtw(i − 1, j) == mindtw then
            i = i − 1
        else if dtw(i, j − 1) == mindtw then
            j = j − 1
        else
            j = j − 1 ;  i = i − 1
        end if
        path.add((i, j))
    end if
end while
return path;
```

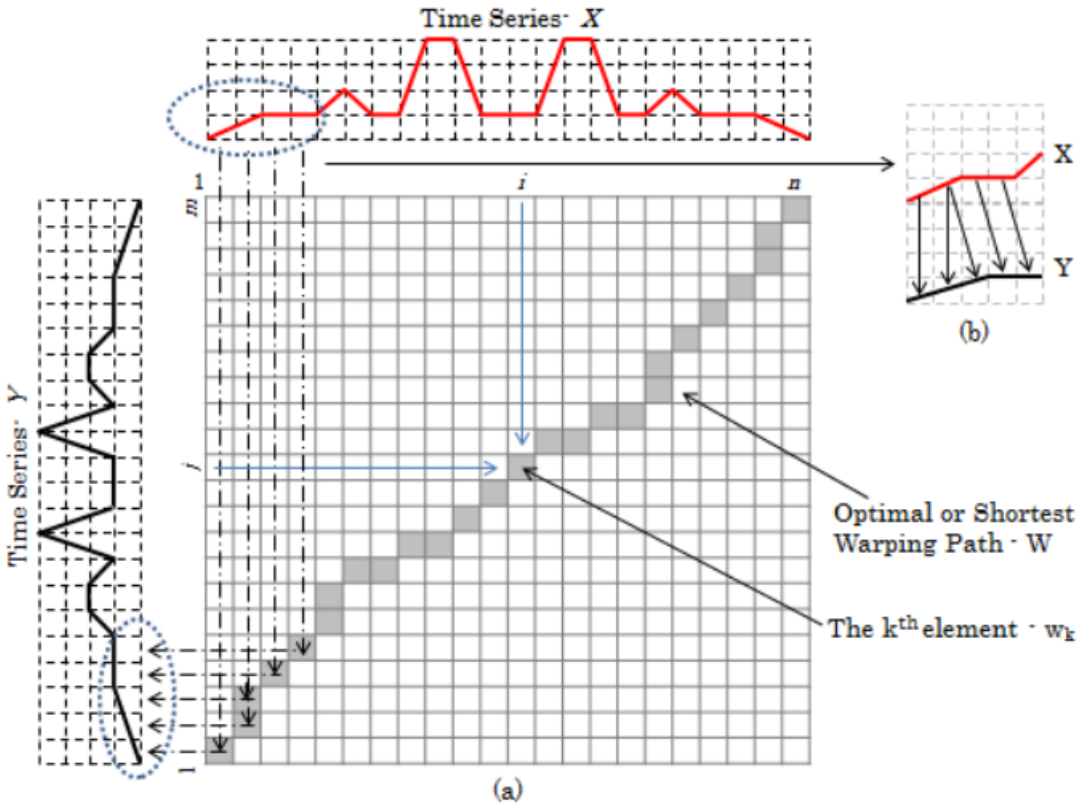


Figure 2.3: 동적 시간 정합 예시 그림 [55]

2.2 관련 연구

2.2.1 다시점 시계열 예측과 딥러닝 모델

전통적인 시계열 예측은 해당 분야의 전문가들의 지식과 경험에 기반한 자기회귀 (Auto-regressive) 모델 [10, 2], 지수평활(exponential smoothing) 모델 [21, 58] 등의 모수적 (parametric) 모델을 주로 활용해왔다. 현대에는 머신러닝이나 딥러닝을 이용하여 순수하게 데이터에 기반한 방법으로 시계열 데이터의 시간적 역학(temporal dynamics)를 배우는 비모수적 모델들 [1] 이 주로 활용되는 추세이다.[38] 에서는 다시점 예측 시 예측 방법에 따라 반복적 예측법(Iterative method)과 직접적 예측법(Direct Method)을 나누었는데 [64] 에서는 이러한 예측 방법의 차이가 곧 성능에 큰 영향을 줄 수 있음을 지적하였다.

반복적 예측법 [6] 은 기본적으로는 과거의 예측을 다시 미래의 예측을 위한 입력으로 사용하는 자귀 회귀적(Autoregressive) 방식을 채택한다. 이는 단시점 예측을 반복적으로 수행하는 것으로 볼 수 있다. 가장 기본적인 RNN 계열 모델들인 LSTM [24], GRU [14]나 과거의 시계열 데이터 분포를 학습하여 미래의 확률 분포를 예측하는 모델로 수요 예측 분야에서 획기적인 성과를 거둔 DeepAR [47] 등을 그 예로 들 수 있다. 그림 2.4에 반복적 예측법의 개념이 설명 되어 있다.

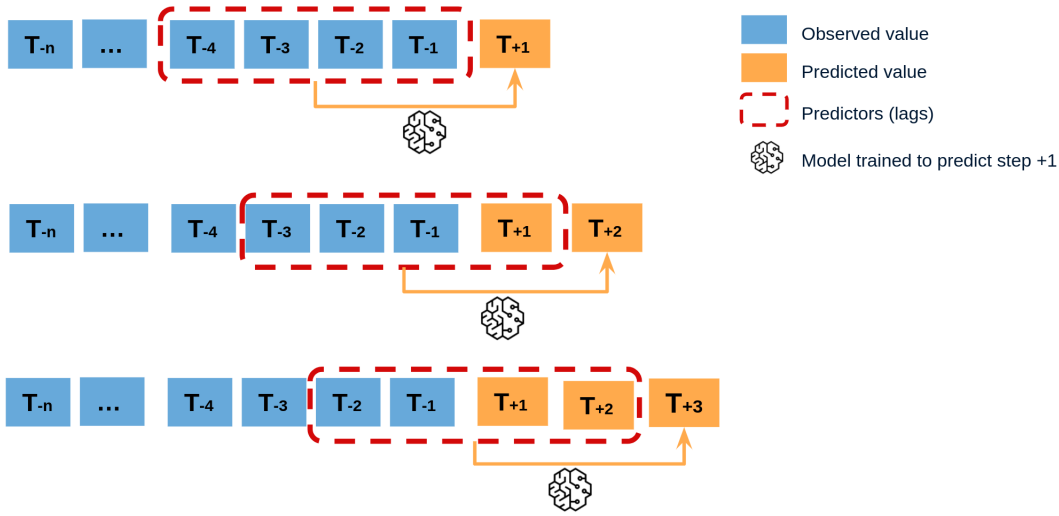


Figure 2.4: 반복적 예측법의 개념 그림 [46]

직접적 예측법 [12]은 다시점의 예측을 한번에 수행한다. 주로 sequence-to-sequence 구조를 사용하며 인코더(Encoder) 에서 과거의 중요한 정보들을 요약하고 디코더(Decoder)에서 그 정보들을 결합하여 예측을 하는 방식이라고 할 수 있다. [64]에서는 훨씬 더 간단한 방법으로 목표 예측 길이와 동일한 고정된 길이의 벡터를 바로 산출하기도 한다. 그림 2.5에 반복적 예측법의 개념이 설명 되어 있다.

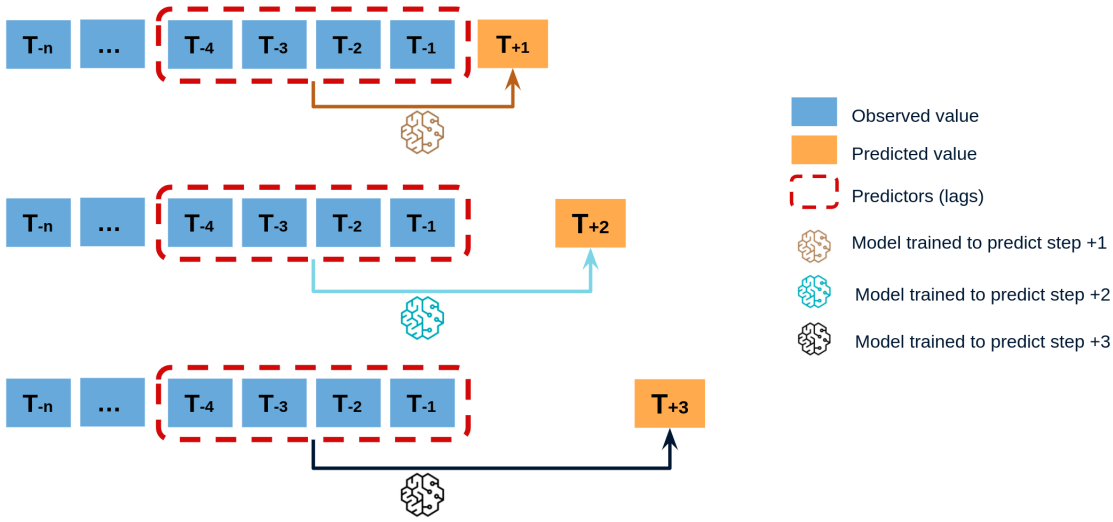


Figure 2.5: 직접적 예측법의 개념 그림 [46]

반복적 예측법은 자기 회귀적인 특성에 의해 직접적 예측법에 비해 더 작은 분산을 갖지만 예측 길이가 늘어날 수록 오차가 누적되어 더 큰 오차의 원인이 되는 한계 또한 보인다. 결론적으로는 예측 성능이 매우 뛰어난 단시점 예측 모델이 존재하고 예측 시점이 적다면 반복적 예측법을 사용해도 좋지만 일반적인 상황에서는 주로 긴 예측 시점을 갖는 문제에 관심이 많기 때문에 직접 예측법이 더 널리 쓰이는 추세이다.

2.2.2 Transformer와 비(非) Transformer 시계열 예측 모델

시계열 예측을 위한 딥러닝 모델로는 Recurrent Neural Network(RNN) [53], Temporal Convolutional Network(TCN) [3] 등이 대표적이고 특히, 자연어 처리, 발화 인식, 동작 분석 등의 인공지능 응용 분야에서 매우 성공적인 결과를 보인 트랜스포머(Transformer) 모델 [57] 이 시계열 분석 및 예측에서도 두각을 나타내고 있다. 이후로 기본적인 트랜스포머 모델을 발전시킨 수 많은 연구들이 진행됐다. 기본 트랜스포머 모델의 셀프 어텐션(self attention) 기법은 2차(Quadratic) 시간/메모리 복잡도를 지녀

예측 시점의 길이가 길어질 수록 계산 복잡도 상의 한계를 보였고 자귀회귀적 방법을 사용하는 디코더(decoder)는 오차 누적 문제의 원인이 됐다.

Informer [67]는 셀프 어텐션 계산 시 ProbSparse 방식을 이용해 계산 복잡도를 $O(LlogL)$로 감소시켰고, Pyraformer [35]는 피라미드형 어텐션으로 시계열 간의 계층적 역학 관계를 포착하여 $O(L)$의 시간 및 공간 복잡도를 달성하였다. 또한 Autoformer [59]는 계절성과 추세성의 분해를 통해 성능 제고를 꾀했고 FEDformer [68]는 여기서 한 걸음 더 나아가 다양한 커널 사이즈의 이동평균 커널에서 추출한 추세 성분들을 합성하는 전문적인 전략을 제시하기도 하였다. 앞서 언급한 모델은 모두 디코더가 직접적 예측법으로 예측을 진행한다.

그런데 최근 들어서는 트랜스포머 계열 모델이 아닌 모델들이 시계열 예측 연구에서 SOTA를 달성하는 일이 잦아지고 있다. 시계열에 특화된 구조 없이 이중 잔차 블록(doubly residual block) 만으로 M4 데이터셋 [37]에 SOTA를 달성한 NBeats [44], 홀수번째와 짝수번째 데이터들의 분리 및 결합을 반복하여 수용야를 비약적으로 확장시킨 TCN 기반의 SCInet [34], 시계열의 추세 부분을 분리한 후, 단 한 층의 선형 인공 신경망 네트워크로 다양한 시계열 데이터 셋에서 SOTA를 석권하고 있는 DLinear [64] 등을 들 수 있다. 또한 RTnet [52] , TS2VEC [62]등 Contrastive learning 을 시계열 예측에 이용한 모델들도 활발히 연구가 진행중이다.

특히 DLinear에서는 트랜스포머 계열 모델의 높은 성능이 multi-head 셀프 어텐션 기법보다는 직접 예측법에서 기인한 것이라고 주장하였다. 트랜스포머는 트랜스포머는 긴 열(sequence)의 각 원소간 의미적 관계(semantic relationship)를 추출하는 것에는 능하다. 그런데 단순 수 데이터(numerical data)에는 의미적 관계가 있다고 보기 어려우며 순서적 정보가 가장 중요한 정보인데 셀프 어텐션을 사용하며 시간 및 순서 정보의 손실이 일어나게 되고 따라서 복잡한 트랜스포머 모델을 사용하는 것보다 단순한 모델

에 직접적 예측법을 이용하면 좋은 예측 성능을 얻을 수 있음을 실험을 통해 보이기도 하였다.

2.2.3 동적 시간 정합 기반 손실함수 연구

동적 시간 정합을 딥러닝 모델의 손실함수로 사용하기 위해 다양한 연구가 진행되어 왔다. [16]에서는 동적 시간 정합이 시계열의 특징을 잘 파악하는 유사도 측정 지표임에 착안한 후, soft min과 global alignment kernel 의 개념을 이용하여 미분이 가능한 형태의 손실함수인 soft dtw를 제안하였다. [31]에서는 유클리드 거리 기반 손실함수가 구간 전체의 오류를 최소화하다 보니 시계열의 급격한 변화를 예리하게 포착하지 못한다는 점에 착안하여 soft dtw에 temporal distortion index의 개념을 결합하여 시계열의 모양과 시간적 차이를 동시에 고려할 수 있는 손실함수 DILATE를 제안한 후, DILATE가 유클리드 거리 기반 손실함수보다 급격한 변화를 갖는 시계열에서 예측 성능이 더 좋음을 보였다. 동 연구진은 이후에 이를 확률적 예측 [22]과 잠재 공간 상에서 모양과 시간의 분리 학습을 통해 성능을 고도화 시킨 STRIPE++ [32] 로 확장시키기도 하였다.

이외에도 soft dtw가 음의 값을 가질 수 있으며 시계열이 동일할 때 최소가 되지 않기 때문에 positive definite divergence 가 아니라는 점을 지적하며 이 문제를 해결한 dubbed soft dtw divergence [8], soft dtw가 삼각 부등식을 만족하지 못하기 때문에 이 문제를 해결한 TC-DTW [51], 동적 시간 정합을 손실함수 뿐 만 아니라 인공신경망으로써 사용하여 시계열의 특징을 추출할 수 있도록 만든 DTWNet [11], Gumble softmin [26]을 사용하여 동적 시간 정합의 연산량을 줄이고자 한 GDTW [36]등 많은 연구가 동적 시간 정합 기반의 손실함수를 고도화 시키고 성능을 제고하기 위해 노력해왔다.

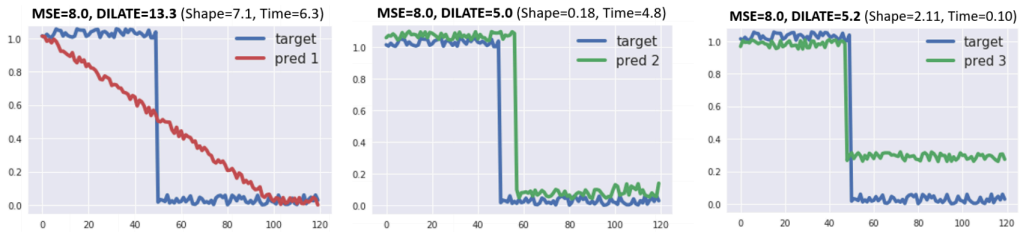
제 3 장 동적 시간 정합 기반 손실함수 개선 기법

3.1 Soft DTW와 DILATE

Soft DTW(이하 SDTW)는 가장 간단한 형태의 동적 시간 정합 기반의 손실함수이다. 예측하고자 하는 목표 시계열을 y^* , 모델이 예측한 시계열을 $\hat{y}$ 라고 하자. 두 시계열 $y^* \in \mathbb{R}^{\tau \times 1}$, $\hat{y} \in \mathbb{R}^{\tau \times 1}$ 의 SDTW 손실함수는 다음 수식 3.1과 같다. $A \in [0,1]^{\tau \times \tau}$ 는 이진 정렬행렬 , $C \in \mathbb{R}^{\tau \times \tau} := \Delta\left(y^*,\ \hat{y}\right)$, ($\Delta$는 l_2 norm)인 지역 비용 행렬, $\mathcal{A}_{\tau,\tau}$ 는 A가 될 수 있는 모든 경우의 집합, 즉 모든 가능한 경로를 나타낸다. 기존 동적 시간 정합 거리는 미분 불가능하다. 따라서 이soft-min 개념을 이용해 미분이 가능하도록 만든 손실함수가 바로 SDTW인 것이다. 평활화의 정도를 조절하는 γ는 하이퍼 파라미터이다.

$$\begin{aligned} DTW_\gamma(\hat{y}, y^*) &:= min^\gamma \left\{\langle A, C\rangle, A \in \mathcal{A}_{\tau,\tau}\right\} \\ &= -\gamma log\left(\sum_{A \in \mathcal{A}_{\tau,\tau}} exp\left(-\frac{\langle A, C\rangle}{\gamma}\right)\right) \end{aligned} \tag{3.1}$$

DILATE(DIstortion Loss including shApe and TimE)은 시계열의 모양과 시점 변화를 더욱 정확하게 포착하기 위해 고안된 인공신경망 모델 학습 역할을 담당하는 손실함수이다. 그림 3.1 은 급격한 변화가 존재하는 계단형 시계열 데이터를 예측하는 상황에서 유클리드 거리인 MSE 가 같더라도 예측 모양이 달라질 수 있음을 보이고 DILATE의 목표는 시계열의 모양과 시간을 동시에 잘 예측하는 것임을 설명한다. 파란색 선은 목표 값, 빨간색 선과 초록색 선은 예측 값을 의미한다.



(a) Non informative prediction (b) Correct shape, time delay (c) Correct time, inaccurate shape

Figure 3.1: 시계열의 예측 형태에 따른 비교 그림 [31]

DILATE는 DTW의 탄성적 왜곡에 대한 불변속성 때문에 SDTW가 시간적 차이는 완전히 무시한다는 점을 지적하며 SDTW와 시간항의 결합을 제시한다.

$$\mathcal{L}_{DILATE}\left(\hat{y}, y^*\right) = \alpha \mathcal{L}_{shape}\left(\hat{y}, y^*\right) + (1-\alpha)\, \mathcal{L}_{temporal}\left(\hat{y}, y^*\right) \tag{3.2}$$

수식 3.2 과 같이 DILATE는 모양 항 $\mathcal{L}_{shape}$ 과 시간 항 $\mathcal{L}_{temporal}$ 의 가중평균으로 정의된다. 여기서 모양 항은 SDTW 와 동일하다.

$$\mathcal{L}_{shape} := DTW_{\gamma}(\hat{y}, y^*) \tag{3.3}$$

DILATE의 시간항은 시간적 왜곡 지수(Temporal distortion index, TDI [19])를 기반으로 한다. 시간적 왜곡 지수는 최적 경로 행렬 A^* 과 1차 대각(Fisrt diagonal) 행렬 즉, 동일한 두 시계열로부터 만들어진 지역 비용 행렬 간의 편차를 의미한다. 다시 말해, 목표 시계열과 예측 시계열이 완전히 동일할 때의 최적 경로와 실제 예측과 목표 시계열 간의 최적 경로가 얼마나 차이 나는지를 나타내는 지표이다. 시간적 왜곡 지수 TDI는 수식 3.4과 같다.

$$TDI\left(\hat{y}, y^*\right) := \langle A^*, \Omega \rangle = \left\langle argmin_{A \in \mathcal{A}_{\tau,\tau}} \langle A, C \rangle, \Omega \right\rangle \tag{3.4}$$

여기서 Ω 는 $\tau \times \tau$ 의 크기를 갖는 실수 정방행렬로 $h, j \in [1, \tau]$ 인 h, j에 대해 $\Omega(h, j) = \frac{1}{\tau^2}(h-j)^2$ 이다. 이는 목표 값과 예측치의 차이 정도를 벌칙으로 부과하는 벌칙 행렬 (Penalizing matrix)라고 할 수 있다.

이 때 수식 3.4의 시간적 왜곡 지수 손실함수 또한 미분 불가능하기 때문에 평활기법을 사용하여 미분가능한 평활 근사치로 만든다. 이 때, 3.4은 서로 다른 값 Ω와 C를 포함하기 때문에 SDTW를 정의할 때와 똑같은 방법을 사용할 수는 없다. $A^* = \nabla_C DTW(\hat{y}, y^*)$ 즉, 최적 정렬 경로는 $DTW(\hat{y}, y^*)$의 지역 비용 행렬에 대한 기울기(gradient) 임을 이용하여, A^*의 평활 근사치인 A^*_γ를 다음 수식 3.5과 같이 정의한다.

$$\begin{gathered} A^*_\gamma = \nabla_C DTW_\gamma(\hat{y}, y^*) = \frac{1}{Z}\sum_{A \in \mathcal{A}_{\tau,\tau}} A exp\left(-\frac{\langle A, C\rangle}{\gamma}\right) \\ Z = \sum_{A \in \mathcal{A}_{\tau,\tau}} exp\left(-\frac{\langle A, C\rangle}{\gamma}\right) \end{gathered} \tag{3.5}$$

따라서 $\mathcal{L}_{temporal}$ 은 수식 3.7 으로 정의할 수 있다.

$$\mathcal{L}_{temporal}(\hat{y}, y^*) := \frac{1}{Z}\sum_{A \in \mathcal{A}_{\tau,\tau}} \langle A, \Omega\rangle exp\left(-\frac{\langle A, C\rangle}{\gamma}\right) \tag{3.6}$$

본 연구에서는 동적 시간 정합 기반 손실함수인 SDTW와 DILATE에 동적 시간 정합의 한계를 개선한 연구를 적용하였다.

3.2 Weighted SDTW

기존 동적 시간 정합은 튀틀림 경로를 구할 때에 정렬 될 수 있는 시점 간 차이에 대한 제한이 없기 때문에 매우 먼 시점 간에도 정렬이 되어 실제 시계열의 패턴 정보를 왜곡할 수 있는 한계가 존재한다. Weighted DTW [27]는 동적 시간 정합을 계산할 때 시점 차이가 클 수록 페널티를 부여하여 너무 먼 시점간 정합하지 않도록 하는 방법을 제시하였다. 그림 3.2은 시계열을 군집화하는 문제에서 각각 다른 패턴을 갖는 두 개의

시계열과 정답 시계열 간 동적 시간 정합 결과를 나타낸다. 동적 시간 정합 값 계산 결과 그림 3.2 a의 파란색의 시계열이 b의 분홍색 시계열 보다 동적 시간 정합값이 더욱 작아 ((a) 41.32 (b) 42.82) 빨간색의 목표 시계열과 더 유사하다는 결과가 나왔지만 실제로는 분홍색 시계열이 목표 시계열과 같은 군집의 시계열로, 이러한 왜곡은 너무 먼 시점 간 정합이 가능해 일어난 일이다.

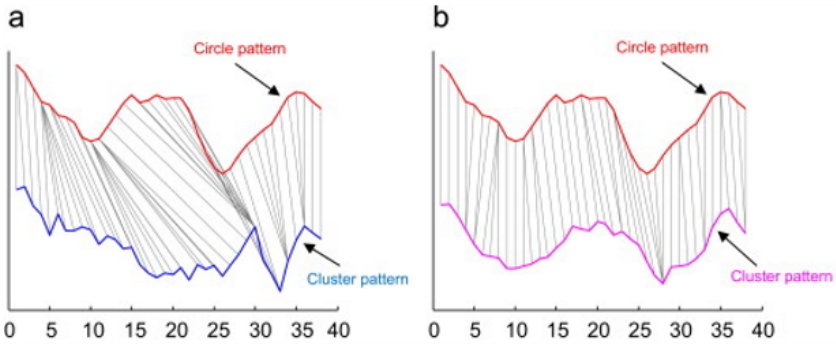


Figure 3.2: 기존 동적 시간 정합 결과 예시 [27]

그림 3.3는 똑같은 시계열을 Weighted DTW로 정합시킨 그림으로, b의 분홍색 시계열이 a의 파란색의 시계열 보다 Weighted DTW값이 더욱 작아((a) 0.16 (b) 0.03) 정답을 그대로 반영한 결과를 산출하였다.

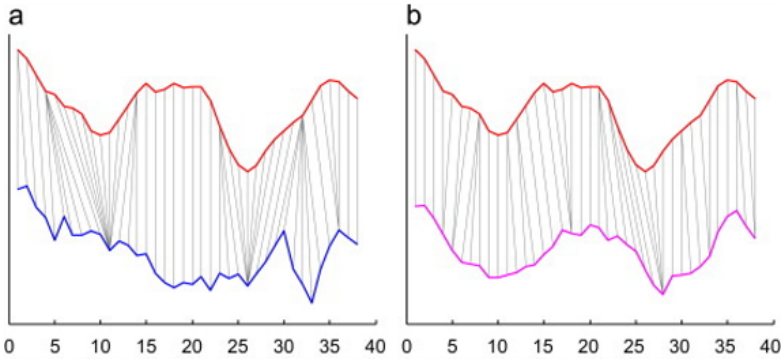


Figure 3.3: Weighted DTW 정합 결과 예시 [27]

본 연구에서는 Weighted DTW를 미분가능한 형태의 SDTW와 결합한 손실함수

Weighted SDTW (이하 WSDTW)를 제안한다. i만큼의 시점 차이가 날 때 부여하는 가중치 $w_i = \left[\frac{w_{max}}{1+exp(-g(i-m_c))}\right]$ 를 활용하여 가중 지역 비용 행렬 C_w 를 제안한다. w_{max} 와 g는 하이퍼 파라미터, m_c는 시계열의 중앙값을 의미한다. 가중 지역 비용 행렬 C_w 은 수식 3.7과 같다.

$$C_w \in \mathbb{R}^{N \times M} : \; w_{|i-j|} c_{i,j} = w_{|i-j|} \times \|x_i - y_j\| \,, \; i \in [1:N], \; j \in [1:M] \tag{3.7}$$

가중 전역 비용 행렬 D를 구하는 pseudo-code는 아래와 같다.

Algorithm 3 전역 비용 행렬 (X, Y, C)

```
n ← |X|
m ← |Y|
wdtw[ ] ← new [N × M]
wdtw(0, 0) ← 0
for i = 1; i ≤ N; i++ do
    wdtw(i, 1) ← wdtw(i − 1, 1) + c(i, 1);
end for
for j = 1; j ≤ M; j++ do
    wdtw(1, j) ← wdtw(1, j − 1) + c(1, j);
end for
for i = 1; i ≤ N; i++ do
    for j = 1; j ≤ M; j++ do
        wdtw(i, j) = min{wdtw(i − 1, j − 1), wdtw(i − 1, j), wdtw(i, j − 1)} +
c(i, j);
    end for
end for
return wdtw
```

SDTW에 계산시 가중 지역 비용 행렬 $C_w \in \mathbb{R}^{\tau \times \tau}$를 대신 사용한 $WSDTW$ 손실함수는 수식 3.8 로 정의한다. $\hat{y}, y^*$ 는 각각 예측 시계열과 목표 시계열, τ는 예측 시계열의 길이, A는 이진 정렬 행렬, $\mathcal{A}_{\tau,\tau}$는 모든 경로 행렬의 집합, γ는 평활 정도를 조절하는 하이퍼 파라미터이다.

$$\begin{aligned} WSDTW(\hat{y}, y^*) &:= WDTW_\gamma(\hat{y}, y^*) \\ &= min^\gamma \left\{ \langle A, C_w \rangle, A \in \mathcal{A}_{\tau,\tau} \right\} \\ &= -\gamma log \left(\sum_{A \in \mathcal{A}_{\tau,\tau}} exp \left(-\frac{\langle A, C_w \rangle}{\gamma} \right) \right) \end{aligned} \tag{3.8}$$

3.3 Shape DILATE

Shape DTW [66]는 기본 동적 시간 정합이 전역적으로는 최적의 해를 찾지만 지역적인 모양까지 세밀하게 잘 정합하지 못하는 한계를 극복하기 위해 각 시점의 주변 시점 정보를 같이 고려하는 방법을 사용하였다. 그림 3.4은 shape dtw의 개념을 나타내고 있다. shape dtw는 입력 시계열에 대하여 부분 시계열에서 모양 기술자 (shape descriptor)를 추출한 뒤, 모양 기술자로 이루어진 새로운 시계열에 대해 동적 시간 정합을 계산하는 단계로 이루어져 있다.

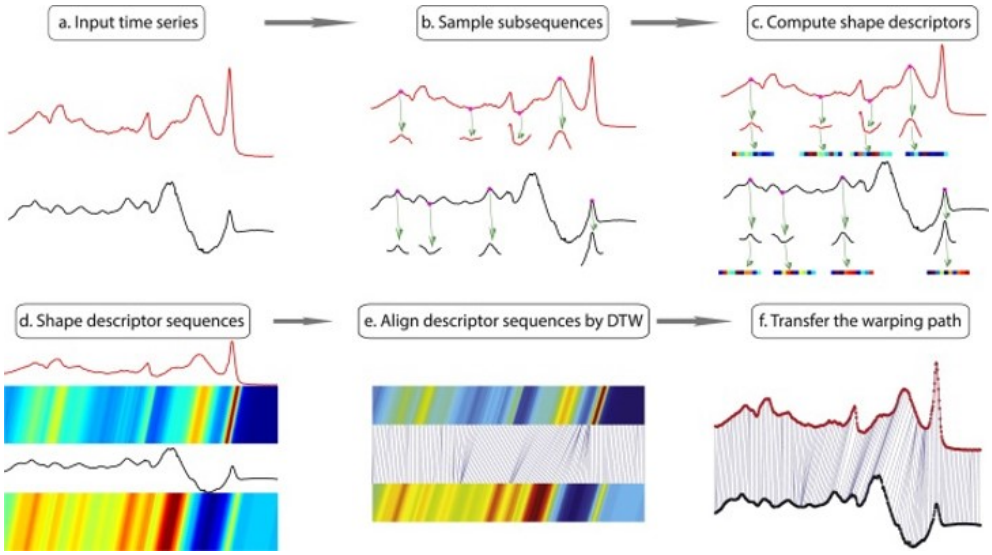


Figure 3.4: Shape DTW 계산 과정 [66]

본 연구에서는 이러한 Shape DTW의 개념을 DILATE에 적용하여 새로운 손실함수인 Shape DILATE를 제안한다. 모양 기술자의 경우 DWT(discrete wavelet transform) [42], 기울기 정보 [29], HOG1D [65] 등을 활용하여 가공할 수 있지만 본 연구에서는 가공되지 않은 길이 l의 부분 시계열을 그대로 모양 기술자로 사용한다. 모양 기술자 $S[y]$는 길의 τ의 시계열 y와 $\lfloor \frac{l}{2} \rfloor < i < \tau - \lfloor \frac{l}{2} \rfloor$ 를 만족하는 모든 자연수 i에 대하여 다음

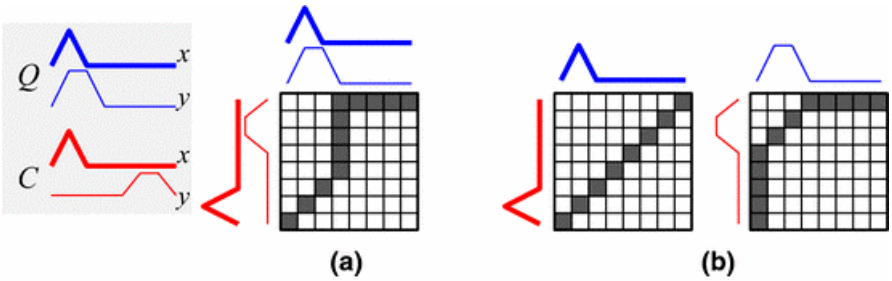
수식 3.9로 나타낸다. $s_i \in \mathbb{R}^{l\times 1}$는 y_i를 중앙값으로 하고 길이 l인 y의 부분 시계열을 의미한다.

$$\begin{gathered} S\left[y_i\right] := s_i := \left[y_{i-\lfloor \frac{l}{2} \rfloor}, ..., y_{i-1}, y_i, y_{i+1}, ..., y_{i+\lfloor \frac{l}{2} \rfloor}\right] \\ S\left[y\right] := s = \left[s_1, ..., s_\tau\right], S\left[y\right] \in \mathbb{R}^{l\times\tau} \end{gathered} \tag{3.9}$$

Shape DILATE은 모양기술자 $S\left[y\right]$를 DILATE에 적용한 손실함수이다. 수식은 3.10 과 같다.

$$\begin{gathered} \mathcal{L}_{ShapeDILATE}\left(\hat{y}, y^*\right) := \mathcal{L}_{DILATE}\left(S\left[\hat{y}\right], S\left[y^*\right]\right) \\ = \alpha\mathcal{L}_{shape}\left(S\left[\hat{y}\right], S\left[y^*\right]\right) + \left(1-\alpha\right)\mathcal{L}_{temporal}\left(S\left[\hat{y}\right], S\left[y^*\right]\right) \end{gathered} \tag{3.10}$$

그런데 Shape DILATE 계산 시, 모양기술자에 대한 다변량 동적 시간 정합 계산이 필요한데 [54] 에 따르면 dependent warping과 independent warping의 두 가지 방식이 존재한다. dependent warping 방식은 동적 시간 정합을 다변량으로 한번에 계산하는 방식이고 independent warping 방식은 각각의 모양기술자에 대한 동적 시간 정합을 개별적으로 계산하고 마지막에 합산하는 방식이다. 그림 3.5 는 이변량 시계열 Q, C 에 대한 동적 시간 정합의 두 가지 연산 방식 차이에 대해 설명한다.



(a) $\mathrm{DTW_D}(Q,C) = \mathrm{DTW}(\{Q_x,Q_y\},\{C_x,C_y\}) = \mathbf{3.2}$

(b) $\mathrm{DTW_I}(Q,C) = \mathrm{DTW}(Q_x,C_x) + \mathrm{DTW}(Q_y,C_y) = \mathbf{2.4}$

Figure 3.5: 다변량 동적 시간 정합 계산 방식에 따른 차이 예시 그림 [54]

그런데 이런 계산 방식의 차이는 분류 및 군집화의 결과 차이로 이어진다. 그림 3.5에서는 A, B, C 세 개의 이변량 시계열에 대해 dependent warping 방식(위쪽)과 independent warping 방식(아래쪽)으로 진행한 군집화 결과를 나타낸다. independent warping 방식은 시간적 차이가 있더라도 모양이 더 비슷한 시계열 A와 B를 더 유사한 시계열로 판단하고 dependent 방식은 모양적 차이가 있더라도 시간적으로 비슷한 시계열 A와 C를 더 유사한 시계열로 판단한다. [54] 에서는 튀틀림 정도와 시간적 지연 크기에 따라 dependent warping이 상대적으로 좋을 수 있지만 independent 방식이 좀 더 강건하다는 사실과 데이터 셋의 특징에 따라 두 방식간 우열의 결과가 다름을 보였다.

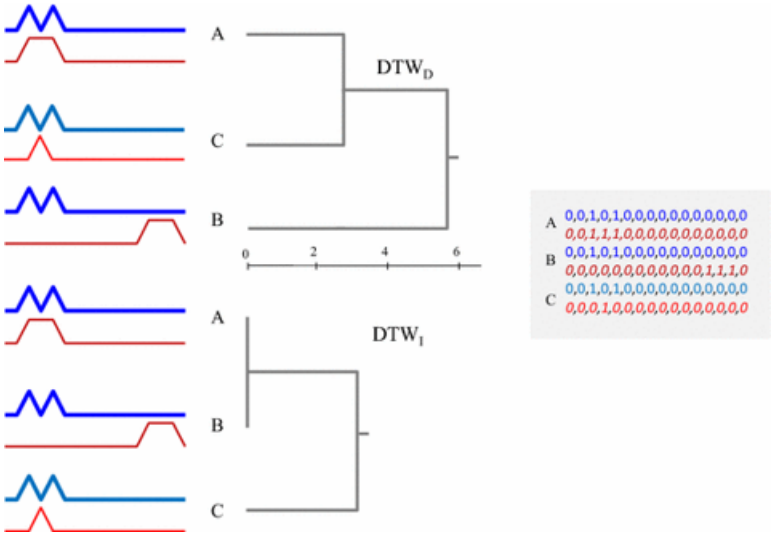


Figure 3.6: 다변량 동적 시간 정합 계산 방식에 따른 군집화 결과 차이 그림 [54]

따라서 본 연구에서는 Shape DILATE 계산 시, dependent warping 방식과 independent warping 방식을 모두 구현하여 먼저 두 방식간 결과의 유의미한 차이가 있는지 확인하고 만약 그렇다면 어떤 방식이 시계열 예측을 위한 손실함수로 더 적절한지 판단해 보는 것을 목적으로 한다.

Shape DILATE 계산시 dependent warping 방식을 사용한 손실함수 ShapeDILATE$_D$, independent warping 방식을 사용한 손실함수 ShapeDILATE$_I$ 는 각각 수식 3.11 과 수식 3.12 로 정의한다. $\hat{s}, s^* \in \mathbb{R}^{l \times \tau}$ 는 각각 예측 부분 시계열, 목표 부분 시계열을 나타낸다.

$$\mathcal{L}_{ShapeDILATE_D}(\hat{y}, y^*) := \mathcal{L}_{DILATE_D}(S[\hat{y}], S[y^*]) = \mathcal{L}_{DILATE}(\hat{s}, s^*) \tag{3.11}$$

$$\mathcal{L}_{ShapeDILATE_I}(\hat{y}, y^*) := \mathcal{L}_{DILATE_I}(S[\hat{y}], S[y^*]) = \sum_{i=1}^{\tau} \mathcal{L}_{DILATE}(\hat{s}_i, s_i^*) \tag{3.12}$$

제 4 장 실험 결과

4.1 모델

본 장에서는 제안된 손실함수의 효과를 검증하기 위해 사용된 세 가지 시계열 예측을 위한 딥러닝 모델에 대해 간단히 살펴본다.

4.1.1 TCN

시계열 예측에 딥러닝 모델이 처음으로 사용될 당시에는 RNN 계열 모델이 많이 사용됐으나 [3]에서 1-d CNN 기반의 모델이 RNN 계열 모델보다 더 예측 성능이 좋음을 보였다. TCN(temporal convolution netowrk)은 미래의 정보가 과거로 흐르지 않는 causal convolution과 수용야를 넓히면서도 연산량의 증가를 불러오지 않는 1차원 dilated convolution 의 장점을 결합한 wavenet [43]을 기반의 모델이지만 네트워크 층(layer) 간 gate activation과 skip connection 을 제거하여 가벼움을 장점으로 한 모델이다. 수용야의 확보를 위해 네트워크의 깊이가 깊어지고 필터(filter) 크기가 커지면서 학습의 안정화를 위해 residual connection을 사용한 것이 특징이다. 그림 4.1은 TCN의 모델 구조를 나타낸다.

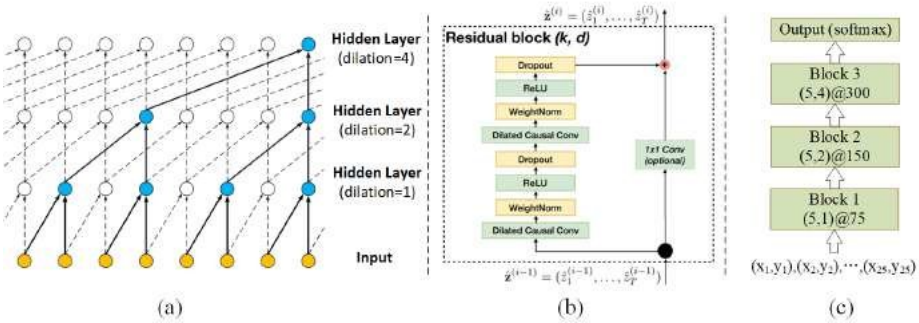


Figure 4.1: TCN 모델 구조 그림[43]

TCN은 RNN 계열 모델과 다르게 병렬화(parallelization)이 가능하고 같은 층에서 필터들 간 가중치를 공유하기에 메모리 사용량이 상대적으로 적다는 특징을 가지면서 넓은 수용야를 통해 좋은 성능을 보여 본 연구의 실험 모델로 선정하였다.

4.1.2 NBeats

NBeats[44]는 통계적 접근법을 전혀 사용하지 않고 M4 데이터 셋에서 SOTA를 차지한 순수한 딥러닝 모델이다. 이 모델은 여러개의 stack으로 이루어져 있는데, 하나의 stack은 다시 여러개의 block으로 이루어져 있다. Doubly residual stacking 기법으로 stack내에 있는 block들이 연결된 구조를 가지며 하나의 block에서 backcast와 forecast를 동시에 진행한다. 이 때 backcast의 residual connection 구조는 입력 부분 중 잘 맞춘 일부 신호를 제거하여 다음 block의 forecast 성능을 높이는 효과를 지닌다. Fully connected layer로 이루어진 block을 특정 함수로 교체하여 주기성이나 추세성을 파악하고 해석이 가능한 예측의 기능도 할 수 있다. 그림 4.2은 NBeats 의 모델 구조를 나타낸다.

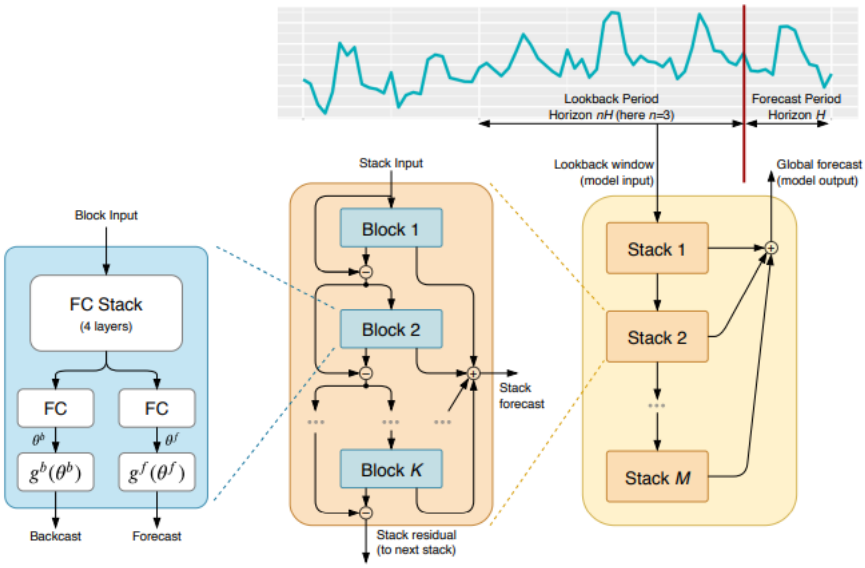


Figure 4.2: NBeats 모델 구조 그림 [44]

Nbeats 모델은 간단하면서도 기본적인 트랜스포머 모델보다 시계열 예측에서 좋은 예측 성능을 보였기에 본 연구의 실험 모델로 선정하였다.

4.1.3 DLinear

DLinear [64]는 시계열을 추세를 담고 있는 부분(이동 평균)과 나머지 부분으로 나눈 후, 각각의 시계열에 과거 정보를 압축할 수 있는 가장 간단한 구조인 1개 층의 선형 네트워크를 통과 시킨 후 그 결과를 다시 결합하여 예측값을 내는 매우 간단한 구조의 딥러닝 모델이다. DLinear는 트랜스포머 모델과 달리 모델 하이퍼 파라미터 튜닝 없이 학습 가능하고 매우 낮은량의 메모리를 사용하고 연산 속도가 매우 빠르다는 특징이 있다. 또한 네트워크의 가중치를 보고 추세나 주기에 관한 해석이 가능하다. 그림 4.3은 DLinear 의 모델 구조를 나타낸다.

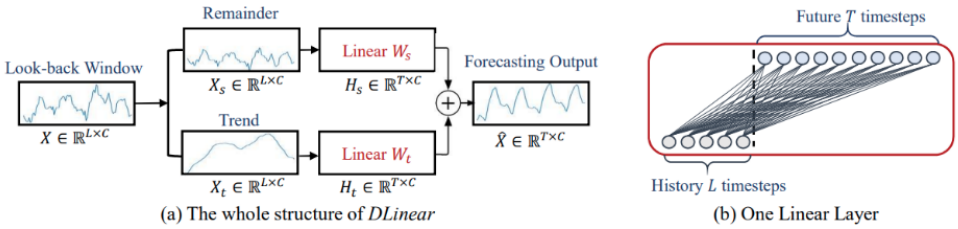


Figure 4.3: DLinear 모델 구조 그림 [64]

DLinear는 본 연구 논문 실험 시점에서 ETT, WTH, ECL 데이터 셋에 대해 SOTA를 달성중인 모델로 매우 간단하면서도 좋은 성능을 지녀 본 연구의 실험 모델로 선정하였다.

4.2 데이터 셋

[52] 에서는 시계열 예측 시에 짚어야 할 시계열의 특성이나 합리적인 가정들을 정리한다. 먼저 시계열 데이터는 내재된 고유의 인과관계를 지닌다는 특성이 있다[9, 10]. 또한 많은 시계열 예측 연구들이 다음과 같은 가정들을 전제로 한다.

(a) 시계열 데이터에는 상한과 하한이 정해져있지 않다 [63].

(b) 시계열 데이터에는 언제나 무작위의 이상치가 존재한다. [60]

(c) 시계열은 부분적으로 자기 회귀적이고 경험적으로 항상 정상성을 만족하는 것은 아니다. [56]

본 연구에서는 시계열 예측 분야 연구에서 주로 많이 쓰는 3개의 데이터 셋인 ETT, WTH, ECL에 급격한 변동성을 가지는 시계열을 대표하기 위해 선정된 BTC 데이터를 포함해 총 4개의 데이터 셋에서 실험을 진행하였다. 4개의 데이터 셋은 앞서 설명한 시계열의 특성과 가정들의 전제를 만족하며 각각 다른 시계열들과 구별되는 고유의 특징을 가지고 있는데 본 장에서는 각 시계열들의 특징을 먼저 분석해본다.

4.2.1 ETT

ETT 데이터는 전기 변압기 온도(Electricity Transformer Temerature)에 대한 데이터로 장기 전력 배치 계획에 있어 중요한 지표 역할을 한다. ETT 데이터는 중국의 도시에서 얻어진 2년간의 기록으로 이루어져 있다. 본 연구에서는 1개 도시에서의 한 시간 단위로 기록된 데이터인 ETTh1 데이터를 사용하고 엔진 오일 온도를 예측한다.

그림 4.4과 그림 4.5는 각각 전체와 부분 ETT 데이터 셋의 목표 예측 변수인 OT(oil temperature)를 나타낸다. ETT 데이터 셋은 변동성은 전역적으로나 지역적으로 매우

큰 변동성을 갖는다. 따라서 엔진 오일 온도(OT)에는 주기성이 존재하지 않을 가능성이 매우 크다. 이는 엔진 오일 온도를 예측할 때, 시간에 대한 임베딩이 큰 역할을 갖지 못함을 암시한다. 또한 전역적으로는 정상성이 없고 부분 입력에 일정한 값이 지속되는 이상치들이 발견되며 자기 회귀성을 갖는다.

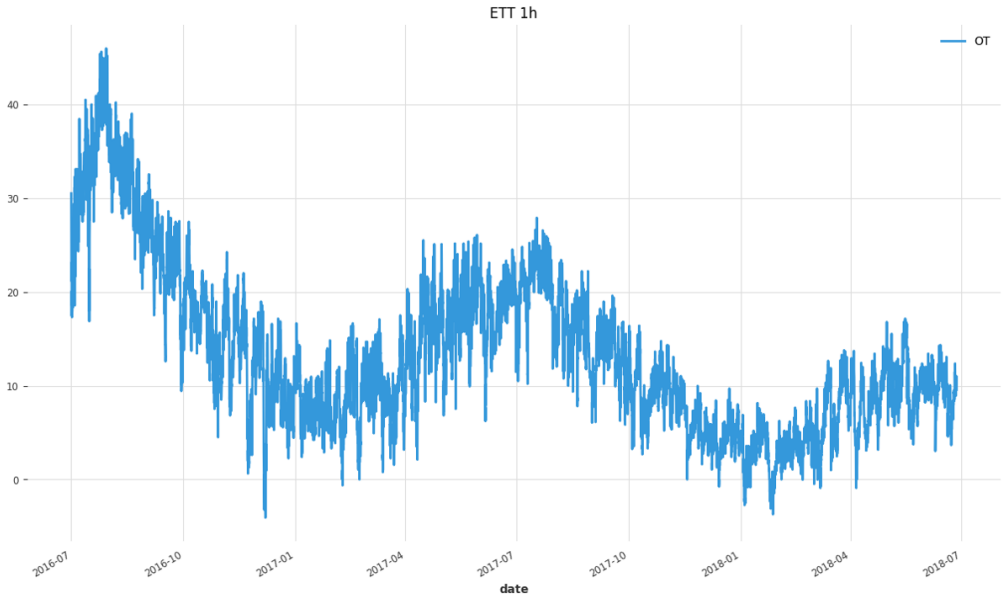


Figure 4.4: 전체 ETT 데이터 셋 그림

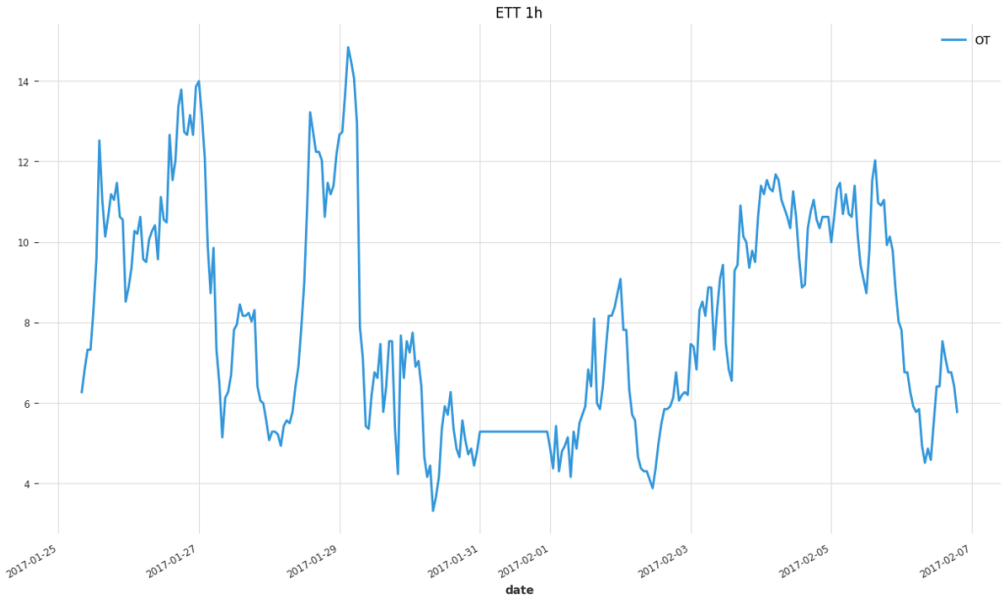


Figure 4.5: 부분 ETT 데이터 셋 그림

4.2.2 WTH

WTH 데이터 셋은 미국 전역 1600여 개 지역에서 2010년부터 2013년까지 수집된 12개의 기후학적 특징들이 기록된 데이터 셋이다. 본 실험에서는 습구 섭씨 온도를 예측한다.

그림 4.6과 그림 4.7는 각각 전체와 부분 WTH 데이터 셋의 목표 예측 변수인 습구 섭씨 온도(wetbulb celcius)를 나타낸다. WTH 데이터 셋은 전역적으로 1년 주기성을 갖는다. 그래서 전역적으로는 정상성을 만족하지만 부분적으로는 통계적 특성이 시간에 따라 변하는 것을 볼 수 있다. 또한 지역적으로 변동성이 크고 일정 값이 지속되는 이상치 또한 발견되며 자기 회귀성을 갖는다. 종합적으로, 습구 섭씨 온도를 예측할 때는, 1년이나 되는 크기의 입력 크기를 설정할 수는 없기 때문에 시간적 임베딩이 필요하다고 할 수 있다.

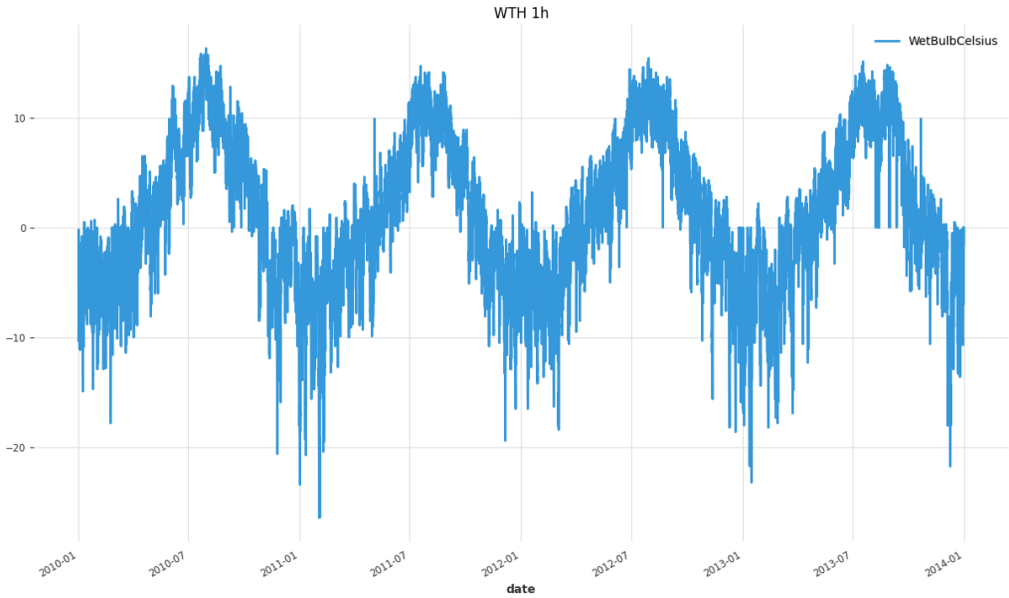


Figure 4.6: 전체 WTH 데이터 셋 그림

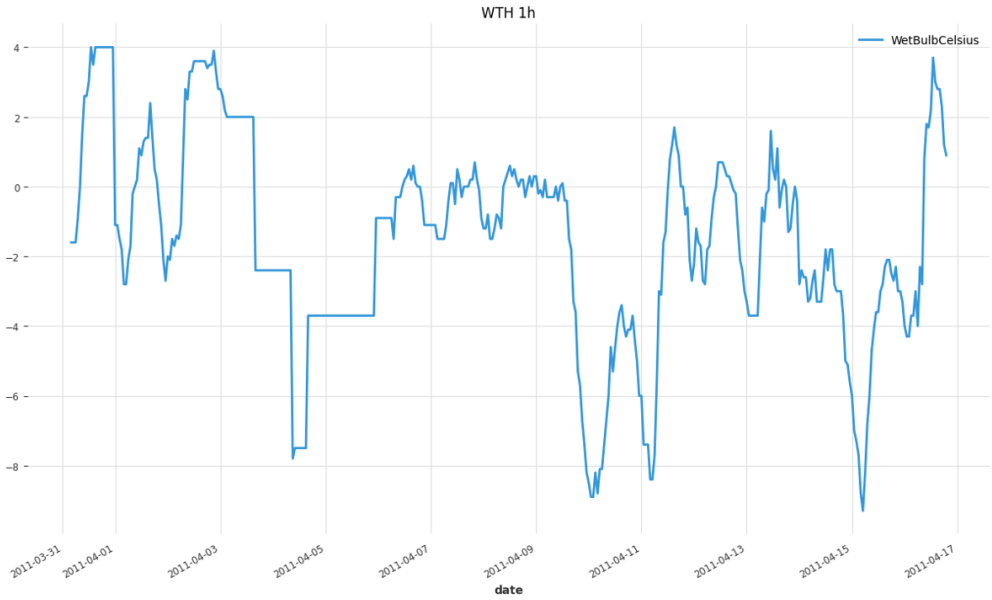


Figure 4.7: 부분 WTH 데이터 셋 그림

4.2.3 ECL

ECL 데이터는 전력 소비(Electricity consumption)에 대한 데이터로 2012년부터 2014년까지 320개의 고객에 대해 15분 단위로 기록된 데이터이다. 본 연구에서는 다른 데이터셋과의 형평성을 고려하여 1시간 단위로 변환된 데이터를 사용하고 320번 고객의 전력 소비량을 예측한다.

그림 4.8과 그림 4.9는 각각 전체와 부분 ECL 데이터 셋의 목표 예측 변수인 320번 고객의 전력 소비량 (MT320)을 나타낸다. ECL 데이터 셋은 1주 주기성을 갖기 때문에 전역적으로나 지역적으로 안정적인 분포를 갖는 시계열이라 할 수 있다. 전역, 지역적으로 모두 정상성을 만족한다고 할 수 있고 상대적으로 이상치가 적으며 자기회귀성을 갖는다. 전력 소비량을 예측할 때, 입력 크기가 시계열의 주기성을 다루기 충분하기 때문에 시간적 임베딩이 큰 영향을 미치지 않을 것이라 예상된다.

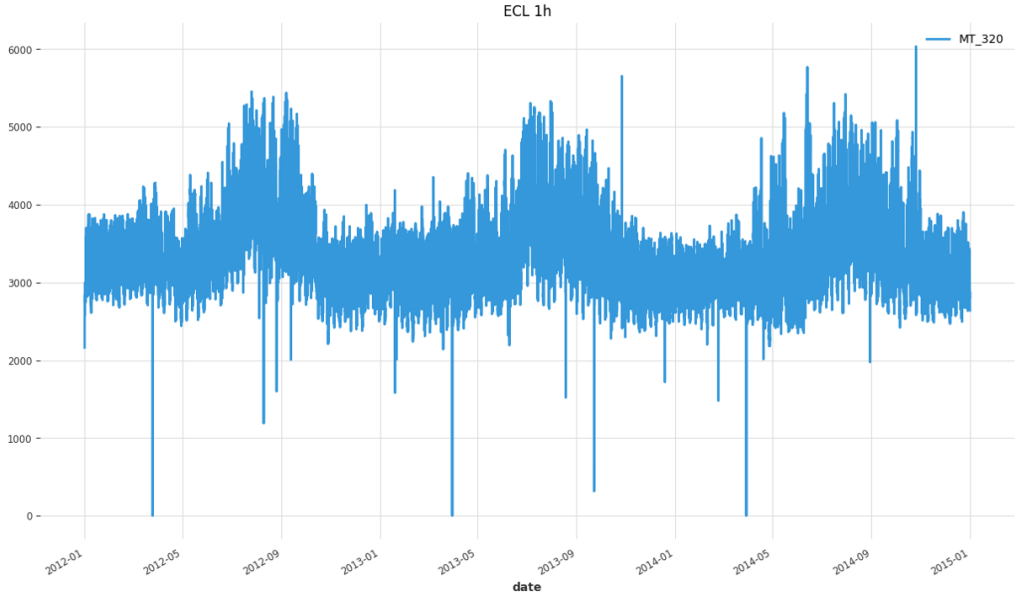


Figure 4.8: 전체 ECL 데이터 셋 그림

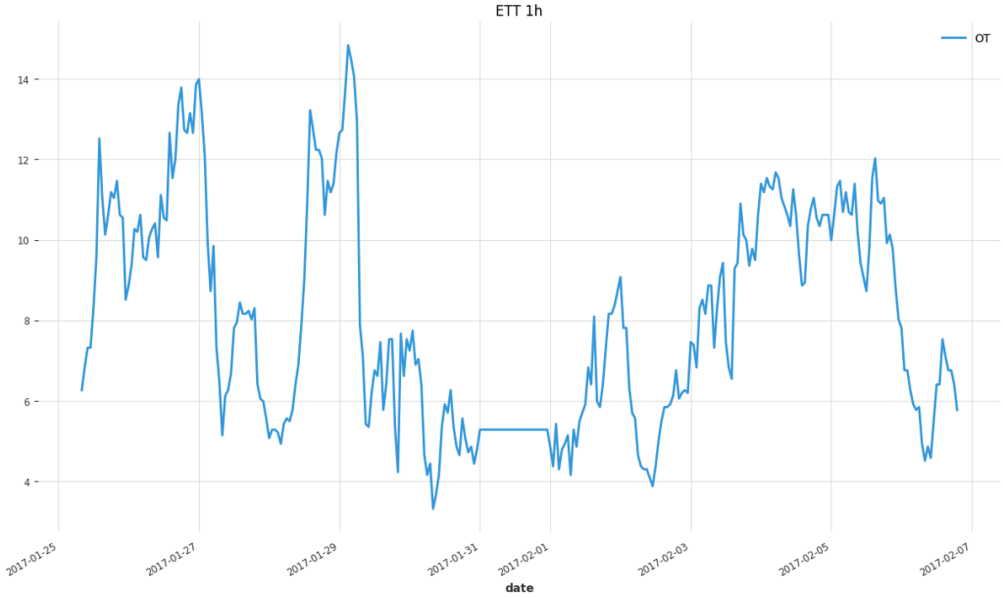


Figure 4.9: 부분 ECL 데이터 셋 그림

4.2.4 BTC

BTC 데이터는 BTC/USDT (비트코인의 테더(Tether) 가격) 데이터로 실험 당시 시점부터 한 시간단위로 5000시점이 되도록 임의로 설정한 기간만큼 [17] 에서 직접 얻었다. 본 연구에서는 비트코인의 가격을 예측한다.

그림 4.10과 그림 4.11는 각각 전체와 부분 BTC 데이터 셋의 목표 예측 변수인 비트코인의 가격을 나타낸다. BTC 앞서 살펴본 3개의 데이터 셋과 비교하여 전역적으로 가장 큰 변동성을 갖고 지역적으로는 변화의 폭이 가장 큰 시계열 데이터라고 할 수 있으며 자기 회귀성을 갖는다. 일정한 값을 가지는 등의 이상치는 발견되지 않았다.

Figure 4.10: 전체 BTC 데이터 셋 그림

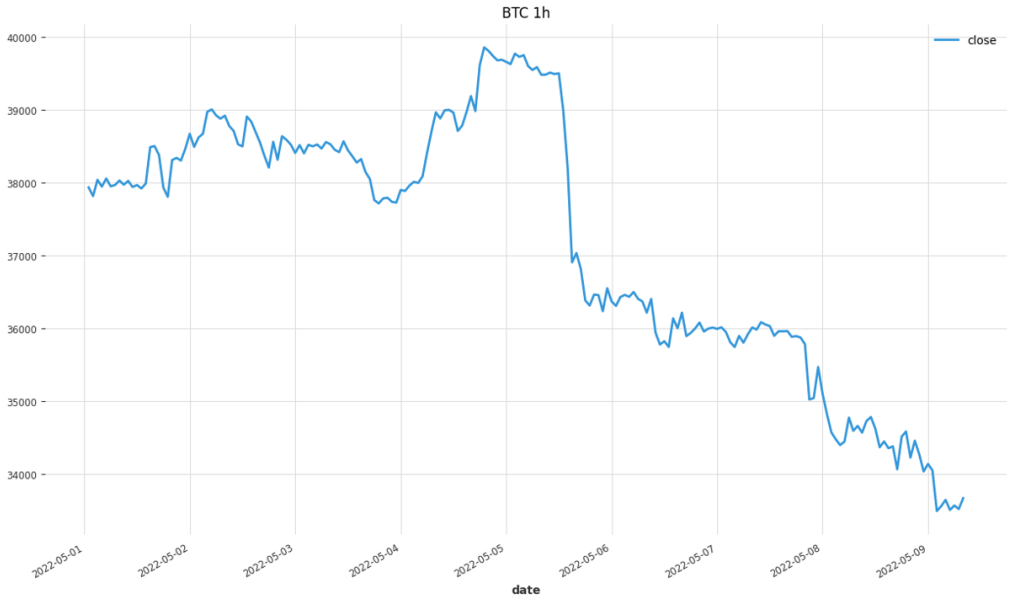


Figure 4.11: 부분 BTC 데이터 셋 그림

이렇게 실험 데이터 셋의 특징을 자세히 알아본 이유는, 시계열의 특성을 충분히 이해하지 않은 상태로 실험을 진행하게 되면 혹여 추후 또 다른 고유한 특성을 가지는 시계열에서도 제안된 기법이 좋은 성능을 보일지 보장할 수가 없기 때문이다. 본 실험에서 선정한 4개의 데이터셋은 현실에 존재하는 다양한 유형의 시계열들을 어느 정도 대표할 수 있을 것으로 기대한다.

실험에 사용된 데이터셋들에 대해 요약된 내용이 표 4.1에 제시되어 있다.

Table 4.1: 데이터셋 요약

데이터 셋	ETT	WTH	ECL	BTC
시점 수	17420	35064	26304	5000
시작 날짜	2016/07/01	2010/01/01	2012/01/01	2022/03/28
종료 날짜	2018/06/26	2013/12/31	2014/12/31	2022/10/23
주기성	없음	전역	지역	없음
정상성	없음	전역	전역, 지역	없음
이상치	존재	존재	존재	존재
자기회귀성	존재	존재	존재	존재

4.3 실험 세팅

본 연구에서 실험에 사용한 모든 데이터 셋은 전체 기간에 대하여 각각 6 : 2 : 2 의 비율로 training set, validation set, test set 으로 분할 후, training set과 validation set 은 모델 학습에 사용하고 test set에서 평가를 진행하였다. 예측 기간 τ는 24로 고정하였지만 입력기간은 24 , 48, 72 중 각 데이터 셋과 모델에서의 실험을 통해 성능이 가장 좋은 값으로 설정하였다.

또한, 실험의 편의를 위해 시계열 예측 용 라이브러리인 Darts [23] 를 사용하였다. Darts는 pytorch [45] 및 pytorch lightning [18]을 기반으로 하여 다양한 시계열 데이터를 처리하고 학습시키고 예측할 수 있는 프레임워크와 다양한 시계열 예측 모델을 제공한다. TCN과 NBeats는 Darts에 내장된 모델을 바로 사용하였고 DLinear는 깃헙 (GitHub)에 공개된 소스코드 [15] 를 활용하여 Darts안에서 사용할 수 있도록 구현하였다. 각 예측 모델의 하이퍼 파라미터는 무작위 탐색을 통한 튜닝 과정을 거쳐 최적의 하이퍼 파라미터를 선정했으며 세부 사항은 다음 표 4.2, 4.3, 4.4 에 나와있다.

Table 4.2: TCN 하이퍼 파라미터

하이퍼 파라미터	값
kernel 크기	5
filter 개수	3
drop out	0.2
weight normalization 여부	false

Table 4.3: NBeats 하이퍼 파라미터

하이퍼 파라미터	값
블록 종류	generic
블록 개수	1
stack 개수	30
layer 개수	4

Table 4.4: DLinear 하이퍼 파라미터

하이퍼 파라미터	값
kernel 크기	25
drop out	0.2
채널 수	1

손실함수의 하이퍼 파라미터 α는 실험을 통해 좋은 성능을 내는 파라미터로 선정하였다. 또한 학습은 배치 크기(batch size) 1024로, 500 에폭(epoch) 동안 진행하였으며 MSE로 학습 시 학습률(Learning rate)를 0.001, 그 외 동적 시간 정합 기반 손실함수로 학습 시 학습률은 0.01로 설정했다. 옵티마이저(opimizer)는 Adam [30] 을 사용하였고 학습 조기 종료(Early stopping) 기법을 사용하여 5 에폭 동안 검증 데이터 셋의 loss가 감소하지 않는다면 학습을 조기에 멈출 수 있도록하여 과적합을 방지하였다.

평가 지표는 세 가지를 사용한다. MSE로 각 시점의 절대적 차이를 평가하고 DTW(평가지표로의 활용을 구분하기 위해 DTW로 표기한다.)로 시계열의 모양적 차이를 평가하며 TDI로 시계열 간 시간적 차이를 평가한다.

4.4 실험 결과

4.4.1 정량적 평가

제안 기법을 ETT, WTH, ECL, BTC 총 4개의 데이터 셋에서 실험한 결과는 각각 표 4.5, 4.6, 4.7, 4.8 과 같다. 이 표들은 각 데이터 셋에 대해 TCN, NBeats, DLinear 총 3개의 모델과 MSE, DILATE, WSDTW, ShapeDILATE$_D$, ShapeDILATE$_I$ 총 5개의 손실함수로 학습 시킨 후 test set에 대한 평가지표인 MSE, DTW, TDI 값에 대한 표이다. 모든 평가지표에 대한 값은 낮을 수록 잘 예측했음을 의미한다. 각 평가지표에서 가장 작은 값은 파란색 굵은 숫자로 표현하고 두번째로 작은 값은 일반 굵기의 파란 숫자로 표현했다.

먼저 ETT 데이터 셋의 경우, 표 4.5를 보면 TCN과 Dlinear 모델에서는 ShapeDILATE$_I$가 세 지표 모두에서 가장 좋은 성능을 보이거나 그에 준하는 성능을 보였다. Nbeats 모델의 경우 dilate가 dtw와 tdi에서 최고 성능을 보였으나 ShapeDILATE$_I$가 같은 두 지표에서 두번째로 높은 성능을 보였다. WSDTW의 경우, 세 모델 모두에서 MSE(손실함수)로 학습했을 때보다 더 좋거나 낮은 MSE(평가지표)를 달성한 것을 볼 수 있다.

표 4.6의 WTH 실험 결과를 보면 세 가지 모델에서 제안된 손실함수인 WSDTW, ShapeDILATE$_D$, ShapeDILATE$_I$가 기존 손실함수인 MSE와 DILATE에 비해 DTW와 TDI가 대체적으로 낮았다. WTH 데이터 셋에서도 MSE(손실함수)로 학습시켰을 때 MSE(평가지표) 값이 가장 낮거나 두번째로 낮았는데 WSDTW는 동적 정합 기반 손실함수 임에도 거의 비슷한 성능을 보인다.

ECL 데이터 셋 결과는 표 4.7에 있다. TCN 모델에서는 WSDTW와 DILATE가, Nbeats와 DLinear 모델에서는 WSDTW와 ShapeDILATE$_I$가 대체로 좋은 성능을 보였다. 특히, WSDTW로 학습한 모델이 세 모델에서 모두 가장 낮은 MSE를 기록했다.

마지막으로 BTC 데이터 셋의 결과인 표 4.8 보면, 모든 모델에서 제안한 3개의

손실함수들이 가장 낮거나 두번째로 낮은 값들을 보였다.BTC 데이터 셋에서는 다른 데이터 셋에 비해 제안 손실함수가 기존 손실함수에 비해 좋음을 상대적으로 명확하게 확인할 수 있었다.

Table 4.5: ETT 데이터셋 실험 결과

데이터 셋	ETT														
모델	TCN					Nbeats					DLinear				
손실함수	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$
MSE	0.0806	0.067	0.0805	**0.0664**	0.067	0.0486	0.0521	**0.0459**	0.0544	0.0522	**0.0455**	0.0497	0.0458	0.0488	0.0484
DTW	1.0111	0.9026	1.1003	**0.898**	0.904	0.7764	**0.7429**	0.767	0.7434	0.7396	0.7532	0.7141	0.7641	**0.7131**	0.7319
TDI	**1.9736**	2.3272	2.5049	2.3543	2.3148	2.2365	**1.6321**	2.25	1.633	1.8081	2.2521	1.4629	2.2171	**1.4332**	1.5625

Table 4.6: WTH 데이터셋 실험 결과

데이터 셋	WTH														
모델	TCN					Nbeats					DLinear				
손실함수	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$
MSE	0.1488	0.1502	**0.1483**	0.1496	0.1496	**0.1028**	0.1108	0.112	0.1082	0.1051	**0.1113**	0.1215	0.1123	0.1282	0.1213
DTW	1.1786	1.1771	**1.1742**	1.1754	1.1795	0.9706	0.9313	1.0124	**0.8955**	0.9069	1.0599	0.9723	0.972	0.9967	**0.9663**
TDI	1.3009	1.2748	1.2938	**1.2039**	1.2492	1.124	0.9951	1.1013	1.0713	**0.975**	1.2374	**0.9956**	1.0891	1.3179	1.0353

Table 4.7: ECL 데이터셋 실험 결과

데이터 셋	ECL														
모델	TCN					Nbeats					DLinear				
손실함수	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$
MSE	0.4208	0.4163	**0.4143**	0.4304	0.4215	0.2316	0.2305	**0.2149**	0.23	0.2466	0.3377	0.3897	**0.3192**	0.3742	0.3719
DTW	2.2112	**2.051**	2.1636	2.1613	2.1715	1.4287	**1.271**	1.3376	1.29	1.3341	1.871	1.59	**1.5788**	1.5822	1.6515
TDI	0.7713	**0.6901**	0.7165	0.7232	0.7328	0.3623	0.3084	0.2983	**0.2912**	0.2986	0.6834	0.4752	0.4604	**0.4591**	0.4736

Table 4.8: BTC 데이터셋 실험 결과

데이터 셋	BTC														
모델	TCN					Nbeats					DLinear				
손실함수	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$
MSE	0.0065	0.0057	0.0096	0.0048	**0.0046**	0.0034	0.003	**0.0026**	0.0029	0.0027	0.0021	0.002	0.002	**0.0019**	**0.0019**
DTW	0.3141	0.2952	0.413	0.2602	**0.2538**	0.2161	0.2035	0.1832	0.2023	**0.1615**	0.15	0.15	0.147	0.149	**0.146**
TDI	3.3902	**2.1838**	3.78	2.632	2.19	3.3508	3.3249	3.4493	**3.0693**	3.1266	2.8567	3	**2.7213**	2.8375	3.081

Table 4.9: 모든 실험에서 각 손실함수의 우위 횟수 표

손실함수	MSE	DILATE	WSDTW	SDILATE$_I$	SDILATE$_D$
MSE	3 (2)	0 (1)	6 (2)	1 (2)	2 (4)
DTW	0 (0)	3 (2)	2 (3)	3 (6)	4 (1)
TDI	1 (0)	4 (2)	1 (3)	5 (2)	1 (5)

표 4.9는 모든 실험에서 각 손실함수의 우위 횟수를 나타낸 표로, 각 항목의 값들은 첫번째로 성능이 좋았던 횟수를, 괄호안의 수는 두번째로 성능이 좋았던 횟수를 나타낸다. 모든 데이터셋과 모든 모델에서 그리고 세가지 각기 다른 평가 지표 모두에서 언제나 제안한 손실함수가 우수함을 보이진 못했다. 하지만 표 4.9의 제안된 손실함수가 전체 실험에서 우위를 차지한 횟수를 보면 기존 손실함수가 최선은 아니라는 것을 확인할 수 있다. MSE(손실함수)로 학습한 모델은 DTW와 TDI의 성능에서 강점을 가지지 못한데 반해 WSDTW로 학습한 모델은 MSE(평가지표) 에서도 MSE(손실함수)로 학습한 모델보다 좋은 성능을 보인 횟수가 많으면서 DTW와 TDI의 성능도 확보하였다. 또한 DILATE는 TDI 지표에서 강점을 보였지만, MSE 지표 기준으로는 좋은 성능을 보이지 못했고 ShapeDILATE$_D$, ShapeDILATE$_I$는 MSE 성능을 어느 정도 확보하면서도, DTW나 TDI의 우위 횟수는 DILATE과 비슷하였다.

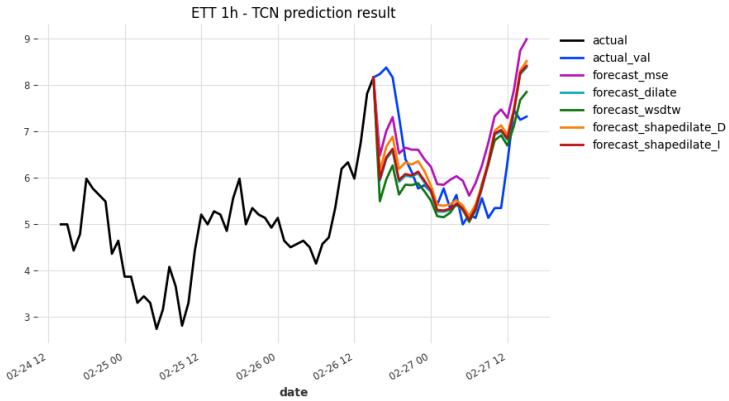
4.4.2 정성적 평가

MSE, DTW, TDI 세 가지 평가지표를 통해 제안된 손실함수의 예측 성능을 알아보았지만 실제로 어떤 형태의 시계열로 예측이 되는지 시각적으로 확인해 DTW나 TDI가 낮은 상황의 의미를 이해해본다. 그림 4.12 - 4.15 은 차례대로 ETT, WTH, ECL, BTC 데이터 셋에서 임의시점을 선정하여 TCN, NBeats, DLinear 모델이 실제 예측한

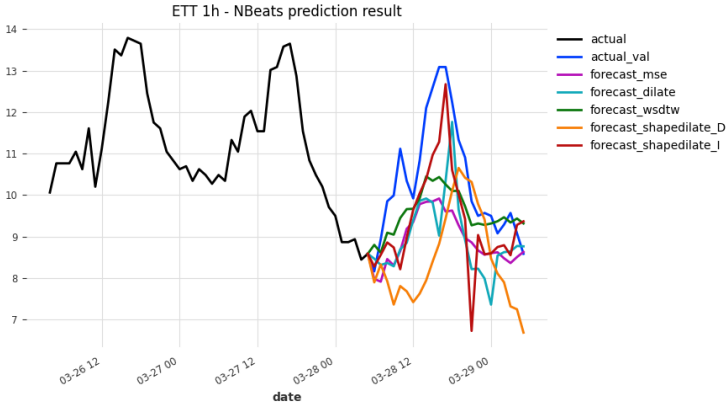
시계열의 그래프이다.

그림 4.12의 (b), (c), 그림 4.13의 (a), (b), 4.14의 (b) 는 예측 시점에 주기성이 존재하는 경우이다. 이 때 세가지 모델 모두 목표 시계열과 유사한 형태의 시계열을 예측하였다. 그런데 MSE나 WSDTW 로 학습시킨 경우보다 ShapeDILATE$_D$ 나 ShapeDILATE$_I$ 로 학습 시킨 경우 산 모양 시계열의 최대 지점을 더욱 정확하게 잡아내는 것을 확인할 수 있다. 4.12의 (c) 에서 DILATE로 학습시킨 경우 뾰족한 모양은 잘 예측하였으나 중간에 감소하는 부분이 있어 모양이나 시점을 정확하게 예측했다고 볼 수 없다. 반면에 ShapeDILATE$_I$ 나 $ShapeDILATE_I$ 로 학습 시킨 경우 모양을 목표 시계열과 더욱 유사하게 예측하였다.

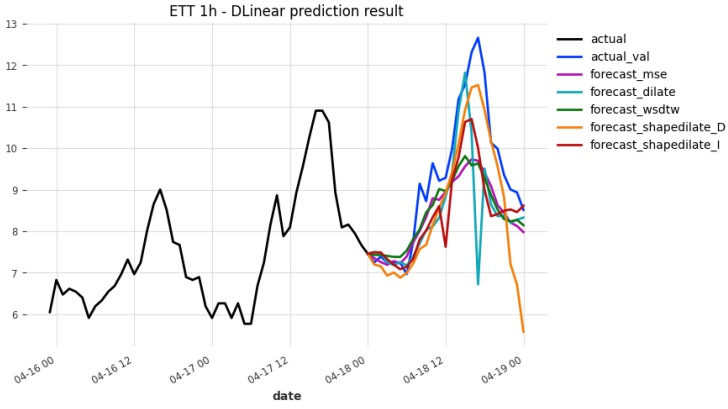
그림 4.13의 (c), 그림 4.14의 (c), 그림 4.15의 (a), (b)는 예측 시점에 급격한 변화가 일어나는 겨웅를 나타낸 그래프이다. 그림 4.13의 (c) 에서 목표 시계열은 골짜기 (valley) 모양을 띄고 있는데 MSE로 학습한 모델이 예측한 시계열의 경우 변화 폭이 매우 낮고 평활한 데에 비해 ShapeDILATE$_D$ 나 ShapeDILATE$_I$ 로 학습 시킨 경우 골짜기 형태의 움푹 들어간 모양을 너으 정도 표현하는것을 볼 수 있다. 4.14의 (c) 과 그림 4.15의 (a), (b) 에도 비슷한 양상을 확인할 수 있다. 예측 시점 직전부터 급격한 변화가 일어날 때, 이전의 시계열 경향보다 직전 변화에 더 큰 영향을 받아 빠르게 변화에 맞춰 예측을 하는 것이 제안 손실함수인 ShapeDILATE$_D$ 나 ShapeDILATE$_I$ 의 특징이라고 할 수 있다.



(a) TCN

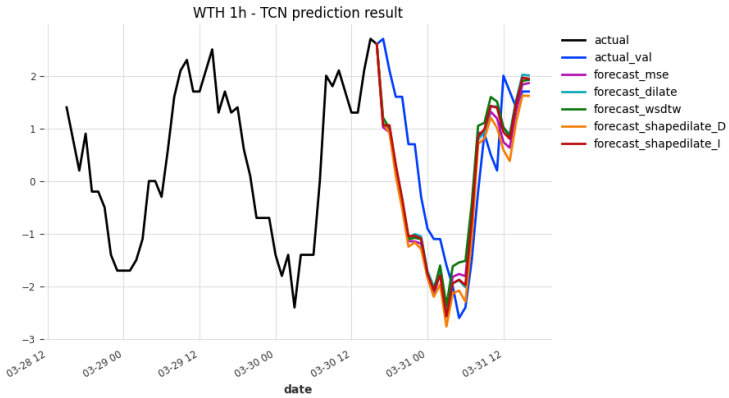


(b) NBeats

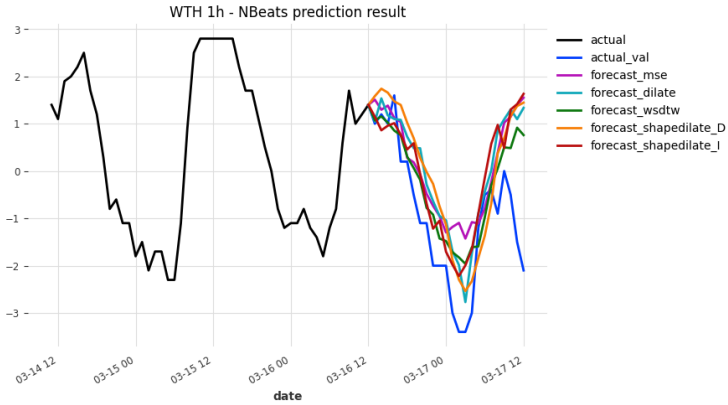


(c) Dlinear

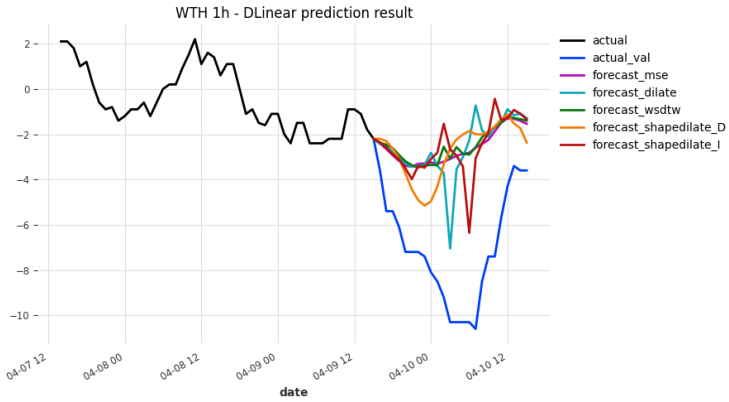
Figure 4.12: ETT 데이터 셋 임의시점 예측 결과



(a) TCN

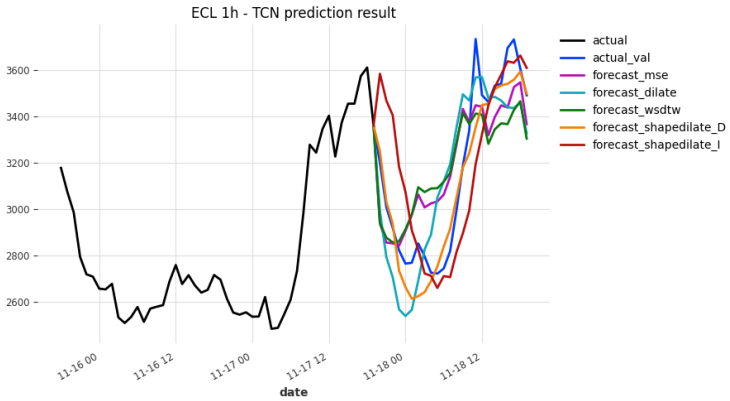


(b) NBeats

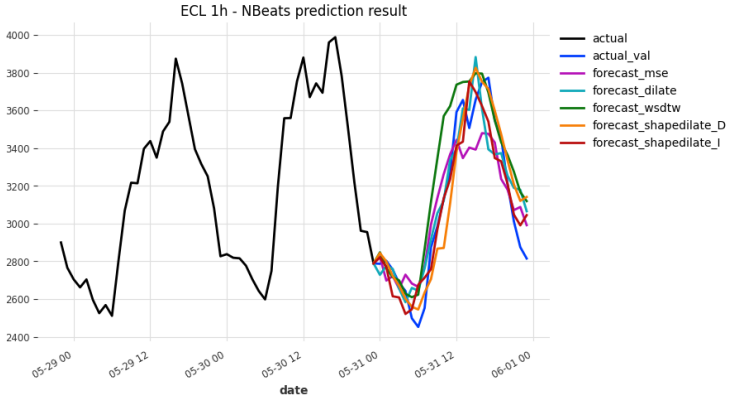


(c) Dlinear

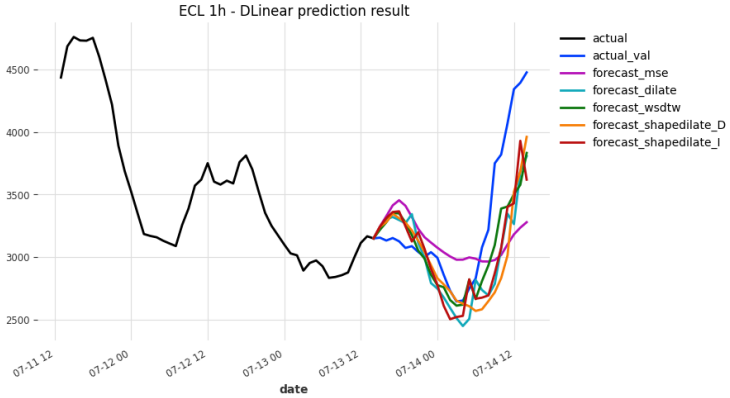
Figure 4.13: WTH 데이터 셋 임의시점 예측 결과



(a) TCN

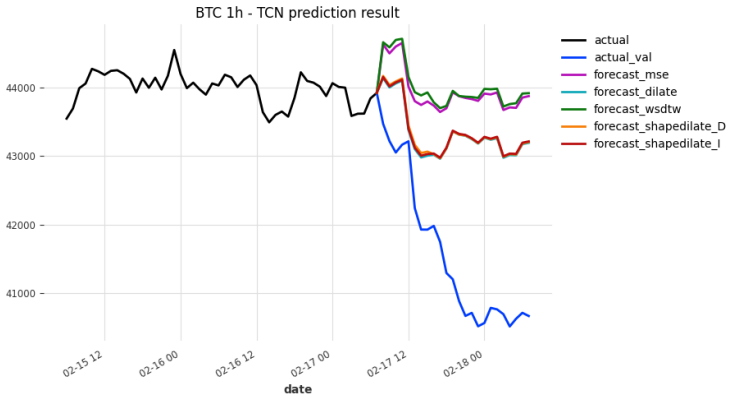


(b) NBeats

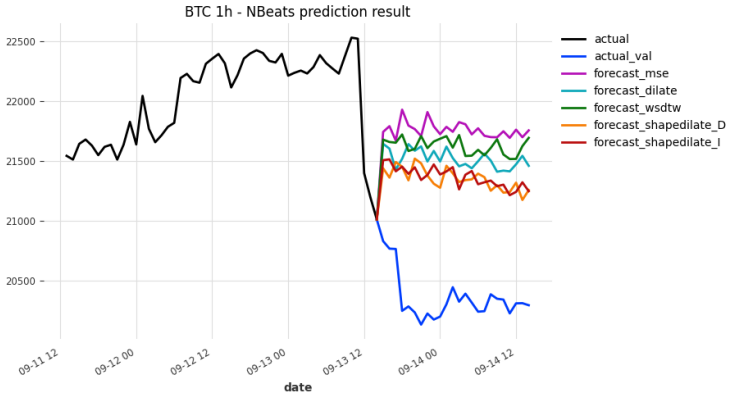


(c) Dlinear

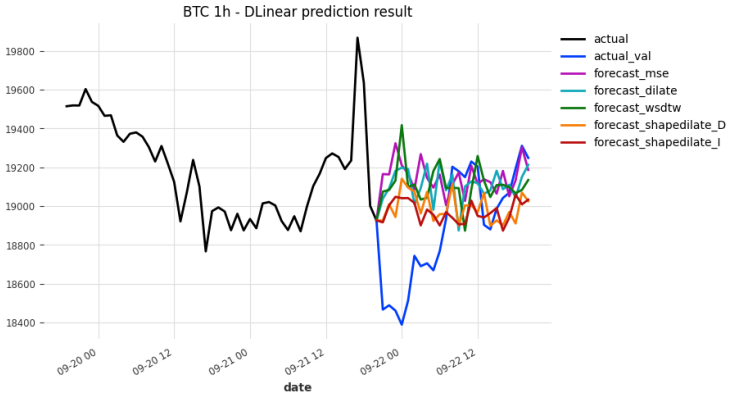
Figure 4.14: ECL 데이터 셋 임의시점 예측 결과



(a) TCN



(b) NBeats



(c) Dlinear

Figure 4.15: BTC 데이터 셋 임의시점 예측 결과

제 5 장 결론

5.1 결론

본 논문에서는 동적 시간 정합 기반의 손실함수를 개선하여 딥러닝 모델이 시계열의 급격한 변화를 더욱 잘 예측할 수 있도록 하는 방법을 제시하였다. 정량적 평가와 정성적 평가를 통해 기존 손실함수인 MSE, DILATE과의 비교를 통해 제안 손실함수인 WSDTW, ShapeDILATE$_I$ 와 ShapeDILATE$_D$ 가 급격한 변화를 갖는 지점이나 시계열의 정확한 모양을 예측하는 데에 유리함을 보였다. MSE를 통해 모든 시점에서의 절대적인 값 차이의 평균을 평가하였고, DTW와 TDI를 추가 평가지표로 사용하여 예측한 시계열과 목표 시계열의 모양이나 변화 시점이 얼마나 유사한지를 동시에 평가하였다. 또한 이 실험이 다양한 특징을 갖는 서로 다른 시계열에서 적용할 수 있는지 확인하기 위해 각각 고유한 특징을 갖는 4개의 시계열 데이터 셋으로 실험을 진행하였고, 모델에 관계없이 성능이 개선됨을 확인하고자 3개의 딥러닝 모델을 사용하여 실험을 진행하였다.

본 논문에서는 시계열 예측 분야에서 주로 쓰이는 3개의 시계열 데이터셋에 더하여 강한 불규칙성을 특징으로 하는 데이터 셋을 추가하여 다양한 특징 유형의 데이터셋을 다루고자 하였지만, 4개의 데이터셋들이 여전히 현실에 존재하는 모든 유형의 시계열을 대표한다고 할 수는 없다. 시계열 데이터들은 이미지 데이터처럼 큰 유형으로 분류하기가 어렵고 각기 고유한 특징을 갖기 때문에 시계열 데이터 셋들을 분류하고 연구와 실험에 사용할 대표 시계열 데이터를 선정하는 것도 하나의 연구 분야가 될 수 있을 것이다. 본 연구에서도 데이터셋 선정에 대한 논의를 깊이 진행하지는 못하였지만, 추후

연구들이 실험 데이터 셋을 선정하고 추가할 때, 특성이 중복되지 않거나 새로운 유형의 시계열 데이터를 선택할 수 있도록 하는데에 도움이 될 것이며, 이러한 연구 결과 들이 축적되어 결국 각 시계열에 특화된 예측 성능 제고로 이어질 것으로 기대된다.

결론적으로 WSDTW는 시점 간 가중치를 부여함으로써 DILATE의 TDI 개념을 쓰지 않고도 시간적 변화 특성을 잘 잡아내면서 MSE 지표에서도 좋은 성능을 보여 MSE의 훌륭한 대안이 될 수 있음을 확인하였다. 또한 급격한 변화나 시계열의 모양을 예리하게 예측할 때에는 주변 시계열을 동시에 활용하여 DILATE의 효과를 극대화한 ShapeDILATE$_I$와 ShapeDILATE$_D$가 기존 DILATE 보다 더 나은 손실함수라고 할 수 있다. 본 연구에서는, MSE가 보편적으로 쓰이는 손실함수이지만, 시계열 예측에 있어서는 시계열의 특성을 더욱 잘 반영할 수 있는 동적 시간 정합 기반의 손실함수가 정성적, 정량적 측면에서 더 효과적임을 보였고 딥러닝 모델의 학습 시에 시계열의 특성을 반영한다는 측면에서 본 연구가 손실함수를 추가적으로 개선하는 추후 연구에 도움이 될 것으로 기대한다.

5.2 향후 연구

동적 시간 정합 기반 손실함수는 시계열의 길이에 따라 연산 시간이 지수적으로 증가하기 때문에 기존 유클리드 거리 기반 손실함수에 비해 연산량이 많아 실제 딥러닝 모델로 학습할 때,시간이 오래 걸린다는 단점이 존재한다. 동적 시간 정합의 연산량을 줄이기 위한 연구 [48, 11, 36]을 손실함수에 적용하여 연산량을 줄일 수 있다면 연구의 실용성을 더욱 제고할 수 있을 것이다.

다른 측면으로는, 동적 시간 정합이 그 자체로 완전한 divergence 가 될 수 없음을 지적한 연구들 [51, 8]이나 ShapeDTW 이후에도 동적 시간 정합 자체의 성능을 개선하려는 연구들 [25, 61]이 존재하는데 해당 연구들이 제안 기법들을 종합적으로 결합한다

면 시계열의 특성을 더욱 잘 반영하면서도 손실함수로서 더 안정적인 동적 시간 정합 기반의 손실함수를 제안할 수 있을 것으로 기대한다.

데이터 셋 선정의 측면에서는, 시계열 데이터 수 부족 문제 해결 방향 제시에 관한 연구로 확장시킬 수도 있을 것이다. 시점 수가 충분히 확보되지 않은 시계열 데이터 셋의 예측 문제에 대해 시계열 데이터의 특성을 유지하는 선에서의 다양한 기법의 데이터 증강 기법을 통해 실제 산업에 적용할 수 있도록 돕는 실용적 측면이 강화된 연구도 진행할 수 있을 것이다.

마지막으로 본 연구에서 단변량 시계열의 비교적 짧은 시점을 예측하는 문제를 정의하였는데 이를 더 긴 시점을 예측하는 문제나 다변량 데이터를 활용하는 문제, 더 나아가 확률적 예측을 하는 문제로 확장시키는 것도 가능할 것이다. 이를 위해 더욱 다양한 데이터 셋을 확보하고 장기 예측과 다변량 예측에 유리한 트랜스포머 등의 모델을 추가하여 추후 연구를 진행할 수 있다.

참고 문헌

[1] N. K. Ahmed, A. F. Atiya, N. E. Gayar, and H. El-Shishiny, *An empirical comparison of machine learning models for time series forecasting*, Econometric reviews, 29 (2010), pp. 594–621.

[2] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, *Stock price prediction using the arima model*, in 2014 UKSim-AMSS 16th international conference on computer modelling and simulation, IEEE, 2014, pp. 106–112.

[3] S. Bai, J. Z. Kolter, and V. Koltun, *An empirical evaluation of generic convolutional and recurrent networks for sequence modeling*, arXiv preprint arXiv:1803.01271, (2018).

[4] C. Banderier and S. Schwer, *Why delannoy numbers?*, Journal of statistical planning and inference, 135 (2005), pp. 40–54.

[5] R. Bellman and R. Kalaba, *On adaptive control processes*, IRE Transactions on Automatic Control, 4 (1959), pp. 1–9.

[6] S. Ben Taieb and R. Hyndman, *Recursive and direct multi-step forecasting: the best of both worlds*, tech. rep., Monash University, Department of Econometrics and Business Statistics, 2012.

[7] D. J. Berndt and J. Clifford, *Using dynamic time warping to find patterns in time series.*, in KDD workshop, vol. 10, Seattle, WA, USA:, 1994, pp. 359–370.

[8] M. Blondel, A. Mensch, and J.-P. Vert, *Differentiable divergences between time series*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2021, pp. 3853–3861.

[9] G. E. Box and G. M. Jenkins, *Some recent advances in forecasting and control*, Journal of the Royal Statistical Society. Series C (Applied Statistics), 17 (1968), pp. 91–109.

[10] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*, John Wiley & Sons, 2015.

[11] X. Cai, T. Xu, J. Yi, J. Huang, and S. Rajasekaran, *Dtwnet: a dynamic time warping network*, Advances in neural information processing systems, 32 (2019).

[12] G. Chevillon, *Direct multi-step estimation and forecasting*, Journal of Economic Surveys, 21 (2007), pp. 746–785.

[13] M. Chui, J. Manyika, M. Miremadi, N. Henke, R. Chung, P. Nel, and S. Malhotra, *Notes from the ai frontier: Insights from hundreds of use cases*, McKinsey Global Institute, (2018), p. 28.

[14] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, *Empirical evaluation of gated recurrent neural networks on sequence modeling*, arXiv preprint arXiv:1412.3555, (2014).

[15] Cure-Lab, *Ltsf-linear/run_longexp.pyatmain · cure − lab/ltsf − linear*, $May2022$.

[16] M. Cuturi and M. Blondel, *Soft-dtw: a differentiable loss function for time-series*, in International conference on machine learning, PMLR, 2017, pp. 894–903.

[17] data and analitics, *cryptodatadownload.com*, 2022.

[18] W. Falcon et al., *Pytorch lightning*, GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning, 3 (2019).

[19] L. Frías-Paredes, F. Mallor, T. León, and M. Gastón-Romeo, *Introducing the temporal distortion index to perform a bidimensional analysis of renewable energy forecast*, Energy, 94 (2016), pp. 180–194.

[20] J. Gao, *Machine learning applications for data center optimization*, (2014).

[21] E. S. Gardner Jr, *Exponential smoothing: The state of the art*, Journal of forecasting, 4 (1985), pp. 1–28.

[22] V. L. Guen and N. Thome, *Probabilistic time series forecasting with structured shape and temporal diversity*, arXiv preprint arXiv:2010.07349, (2020).

[23] J. Herzen, *Time series made easy in python, darts.* `https://unit8co.github.io/darts/`, 2021.

[24] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural computation, 9 (1997), pp. 1735–1780.

[25] J. Y. Hong, S. H. Park, and J.-G. Baek, *Ssdtw: Shape segment dynamic time warping*, Expert Systems with Applications, 150 (2020), p. 113291.

[26] E. Jang, S. Gu, and B. Poole, *Categorical reparameterization with gumbel-softmax*, arXiv preprint arXiv:1611.01144, (2016).

[27] Y.-S. Jeong, M. K. Jeong, and O. A. Omitaomu, *Weighted dynamic time warping for time series classification*, Pattern recognition, 44 (2011), pp. 2231–2240.

[28] E. Keogh and C. A. Ratanamahatana, *Exact indexing of dynamic time warping*, Knowledge and information systems, 7 (2005), pp. 358–386.

[29] E. J. Keogh and M. J. Pazzani, *Derivative dynamic time warping*, in Proceedings of the 2001 SIAM international conference on data mining, SIAM, 2001, pp. 1–11.

[30] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, arXiv preprint arXiv:1412.6980, (2014).

[31] V. Le Guen and N. Thome, *Shape and time distortion loss for training deep time series forecasting models*, Advances in neural information processing systems, 32 (2019).

[32] ——, *Deep time series forecasting with shape and temporal criteria*, IEEE Transactions on Pattern Analysis and Machine Intelligence, (2022).

[33] B. Lim and S. Zohren, *Time-series forecasting with deep learning: a survey*, Philosophical Transactions of the Royal Society A, 379 (2021), p. 20200209.

[34] M. Liu, A. Zeng, Z. Xu, Q. Lai, and Q. Xu, *Time series is a special sequence: Forecasting with sample convolution and interaction*, arXiv preprint arXiv:2106.09305, (2021).

[35] S. Liu, H. Yu, C. Liao, J. Li, W. Lin, A. X. Liu, and S. Dustdar, *Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting*, in International Conference on Learning Representations, 2021.

[36] X. Liu, N. Li, and S.-T. Xia, *Gdtw: A novel differentiable dtw loss for time series tasks*, in ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2021, pp. 2860–2864.

[37] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, *The m4 competition: 100,000 time series and 61 forecasting methods*, International Journal of Forecasting, 36 (2020), pp. 54–74.

[38] M. Marcellino, J. H. Stock, and M. W. Watson, *A comparison of direct and iterated multistep ar methods for forecasting macroeconomic time series*, Journal of econometrics, 135 (2006), pp. 499–526.

[39] M. Müller, *Dynamic time warping*, Information retrieval for music and motion, (2007), pp. 69–84.

[40] M. Müller, H. Mattes, and F. Kurth, *An efficient multiscale approach to audio synchronization.*, in ISMIR, vol. 546, Citeseer, 2006, pp. 192–197.

[41] C. Myers, L. Rabiner, and A. Rosenberg, *Performance tradeoffs in dynamic time warping algorithms for isolated word recognition*, IEEE Transactions on Acoustics, Speech, and Signal Processing, 28 (1980), pp. 623–635.

[42] H. Olkkonen, *Discrete Wavelet Transforms: Biomedical Applications*, BoD–Books on Demand, 2011.

[43] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, *Wavenet: A generative model for raw audio*, arXiv preprint arXiv:1609.03499, (2016).

[44] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, *N-beats: Neural basis expansion analysis for interpretable time series forecasting*, arXiv preprint arXiv:1905.10437, (2019).

[45] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *Pytorch: An imperative style, high-performance deep learning library*, in Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035.

[46] J. A. Rodrigo and J. E. Ortiz, *Skforecast: Time series forecasting with python and scikit-learn*, Feb 2021.

[47] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, *Deepar: Probabilistic forecasting with autoregressive recurrent networks*, International Journal of Forecasting, 36 (2020), pp. 1181–1191.

[48] S. Salvador and P. Chan, *Toward accurate dynamic time warping in linear time and space*, Intelligent Data Analysis, 11 (2007), pp. 561–580.

[49] J. Sen and S. Mehtab, *Accurate stock price forecasting using robust and optimized deep learning models*, in 2021 International Conference on Intelligent Technologies (CONIT), IEEE, 2021, pp. 1–9.

[50] P. Senin, *Dynamic time warping algorithm review*, Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA, 855 (2008), p. 40.

[51] D. S. Shen and M. Chi, *Tc-dtw: Accelerating multivariate dynamic time warping through triangle inequality and point clustering*, Information Sciences, (2022).

[52] L. Shen, Y. Wei, and Y. Wang, *Respecting time series properties makes deep time series forecasting perfect*, arXiv preprint arXiv:2207.10941, (2022).

[53] A. Sherstinsky, *Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network*, Physica D: Nonlinear Phenomena, 404 (2020), p. 132306.

[54] M. Shokoohi-Yekta, B. Hu, H. Jin, J. Wang, and E. Keogh, *Generalizing dtw to the multi-dimensional case requires an adaptive approach*, Data mining and knowledge discovery, 31 (2017), pp. 1–31.

[55] S. Thajchayapong et al., *Generalizability and transferability of incident detection algorithm using dynamic time warping*, in 19th ITS World CongressERTICO-ITS EuropeEuropean CommissionITS AmericaITS Asia-Pacific, 2012.

[56] S. Tonekaboni, D. Eytan, and A. Goldenberg, *Unsupervised representation learning for time series with temporal neighborhood coding*, arXiv preprint arXiv:2106.00750, (2021).

[57] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, *Attention is all you need*, Advances in neural information processing systems, 30 (2017).

[58] P. R. Winters, *Forecasting sales by exponentially weighted moving averages*, Management science, 6 (1960), pp. 324–342.

[59] H. Wu, J. Xu, J. Wang, and M. Long, *Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting*, Advances in Neural Information Processing Systems, 34 (2021), pp. 22419–22430.

[60] J. Xu, H. Wu, J. Wang, and M. Long, *Anomaly transformer: Time series anomaly detection with association discrepancy*, arXiv preprint arXiv:2110.02642, (2021).

[61] J. Yuan, Q. Lin, W. Zhang, and Z. Wang, *Locally slope-based dynamic time warping for time series classification*, in Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 1713–1722.

[62] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, Y. Tong, and B. Xu, *Ts2vec: Towards universal representation of time series*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, 2022, pp. 8980–8987.

[63] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, and B. Xu, *Learning timestamp-level representations for time series with hierarchical contrastive loss*, arXiv e-prints, (2021), pp. arXiv–2106.

[64] A. Zeng, M. Chen, L. Zhang, and Q. Xu, *Are transformers effective for time series forecasting?*, arXiv preprint arXiv:2205.13504, (2022).

[65] J. Zhao and L. Itti, *Classifying time series using local descriptors with hybrid sampling*, IEEE Transactions on Knowledge and Data Engineering, 28 (2015), pp. 623–637.

[66] ——, *shapedtw: Shape dynamic time warping*, Pattern Recognition, 74 (2018), pp. 171–184.

[67] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, *Informer: Beyond efficient transformer for long sequence time-series forecasting*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 11106–11115.

[68] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, *Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting*, arXiv preprint arXiv:2201.12740, (2022).

[69] T. Zonta, C. A. Da Costa, R. da Rosa Righi, M. J. de Lima, E. S. da Trindade, and G. P. Li, *Predictive maintenance in the industry 4.0: A systematic literature review*, Computers & Industrial Engineering, 150 (2020), p. 106889.

Abstract

Improving Time Series Forecasting Performance of Deep Learning Models by Enhancing Dynamic Time Warping based Loss Function

Jaehee Kim
Department of Industrial Engineering
The Graduate School
Seoul National University

Multi-step time series forecasting is a topic that has been studied very actively in various industries and research fields that can obtain historical data that changes over time. Currently, research methodologies that use deep learning models to learn temporal dynamics such as periodicity, trendiness, and irregularity, which are unique characteristics of time series, are common, but there are not many studies on loss functions that evaluate predictions of model and determine learning methods and directions.

In this paper, we present an improved dynamic time warping (DTW)-based loss function for learning deep learning models for time series prediction. We intend to apply a distance-based weighting method (Weighted DTW) and a shape descripting

method (Shape DTW) that considers the surrounding points together, to a differentiable DTW to more accurately predict the shape of the target time series (size and timing of change). We apply the proposed loss function to several deep learning models and real-world datasets with different features, and compare it with Euclidean distance-based loss function and existing DTW-based loss function to show improved predictive performance. In addition, time series prediction was evaluated from various perspectives using various evaluation indicators from a quantitative point of view, and it was confirmed that the loss function presented through qualitative evaluation predicts the rapid change of the time series better, which can sufficiently replace the existing loss function.

Keywords: Multi-step time series forecasting, Dynamic Time Warping, Deep Learning Model, Loss Function, Industrial engineering

Student Number: 2021-27604