TimePoint: Accelerated Time Series Alignment via Self-Supervised Keypoint and Descriptor Learning

Ron Shapira Weber [1] **Shahar Ben Ishay** [1] **Andrey Lavrinenko** [1] **Shahaf E. Finder** [1] **Oren Freifeld** [1]

Abstract

We introduce TimePoint (TP), a self-supervised model for fast and robust time series alignment, trained entirely on synthetic data. Unlike prior foundation models that focus on forecasting, TP targets the critical but underexplored task of alignment by learning generalizable temporal landmarks (keypoints and descriptors). To support this, we propose SynthAlign, a synthetic data generation pipeline using structured waveform composition and CPAB-based time warping with ground-truth correspondences. Despite no access to real data during training, TP achieves zero-shot generalization on 100+ real-world datasets, outperforming DTW variants in both alignment accuracy and efficiency. We position TP as an alignment-centric foundation model, showing that high-quality synthetic data and task-specific pretraining can unlock scalable and transferable representations for time series analysis beyond forecasting. Our code is available at https://github.com/BGU-CS-VIL/TimePoint.

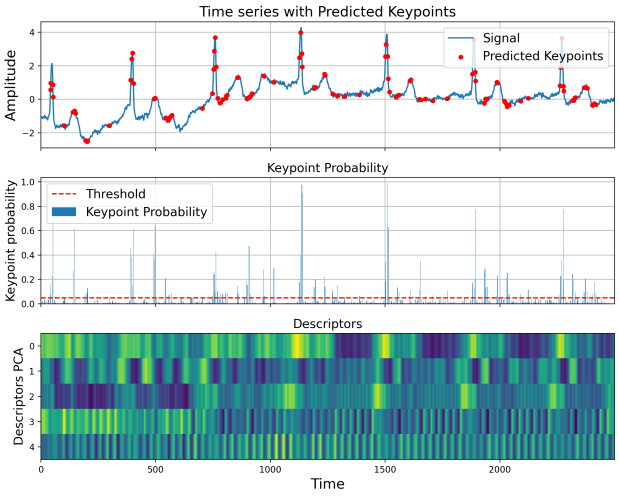


Figure 1. TimePoint (TP): Keypoint Detection and Descriptors on real-world, unseen, ECG data of length 2500 (TP was trained on synthetic data of length 512). Each panel depicts (top-to-bottom) the original signal and predicted keypoints, keypoint probability map, and PCA of the learned descriptors ($D = 256$, using 5 principal components for visualization purposes).

1. Introduction

Time series alignment is a core challenge in structured data analysis, especially when sequences differ in length, sampling rate, or exhibit temporal distortions. While methods like *Dynamic Time Warping (DTW)* offer flexibility, they suffer from $\mathcal{O}(L^2)$ complexity and sensitivity to noise, limiting their use in large-scale or high-throughput applications.

In contrast to recent progress in time-series foundation models focused on forecasting (Ansari et al., 2024; Fu et al., 2024), alignment remains an underexplored but critical task. Inspired by keypoint-based matching in computer vision (DeTone et al., 2018), we introduce *TimePoint* (TP): a self-supervised model that detects keypoints and descriptors in 1D signals to enable *sparse DTW* over salient temporal landmarks. Figure 1 shows TP predicted features and detected KPS for *unseen* ECG data.

To enable self-supervised training, we propose `SynthAlign`, a synthetic dataset generation framework that composes 1D waveforms, computes keypoints (KPs), and applies nonlinear temporal distortions via *Continuous Piecewise Affine Based (CPAB)* transformations (Freifeld et al., 2017), yielding ground-truth correspondences. TP uses a fully convolutional encoder with *Wavelet Convolutional (WTConv)* layers (Finder et al., 2024), enabling efficient inference on long signals. Despite being trained exclusively on synthetic data, TP generalizes across over 100 real-world datasets from the UCR archive (Dau et al., 2019), outperforming classical and differentiable DTW variants in both alignment accuracy and runtime. The entire framework is detailed in Figure 2.

Our contributions are as follows:

*Equal contribution [1]Ben Gurion University of the Negev. Correspondence to: Ron Shapira Weber <ronsha@post.bgu.ac.il>.

Proceedings of the 1st ICML Workshop on Foundation Models for Structured Data, Vancouver, Canada. 2025.

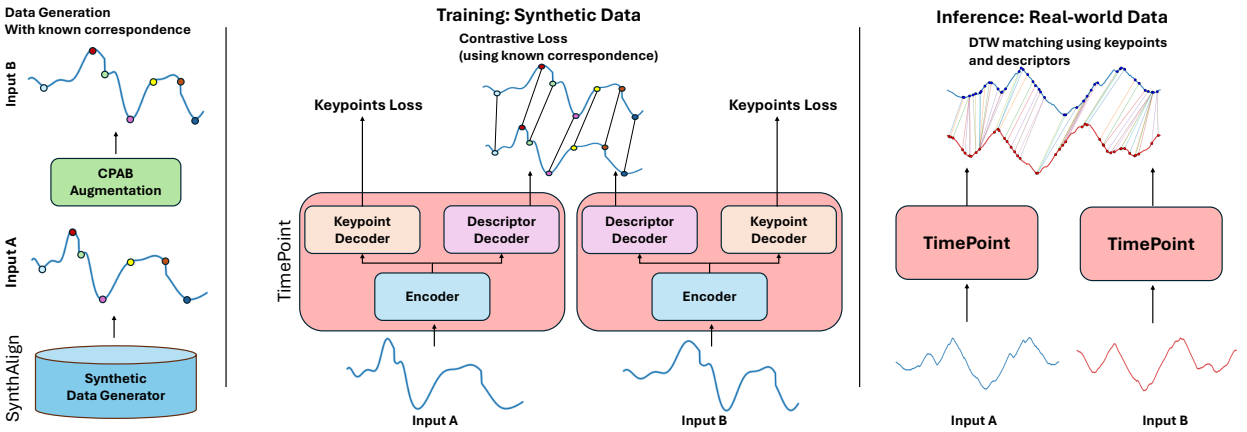


Figure 2. Training and Inference overview. Left: signals and keypoints (KP) are synthetically generated and augmented using CPAB warps (Section 3). Middle: TimePoint (TP) predicts KP location and descriptors using the known correspondence (Section 4). Right: real-world, unseen data pairs are matched using DTW on TP descriptors at KP locations.

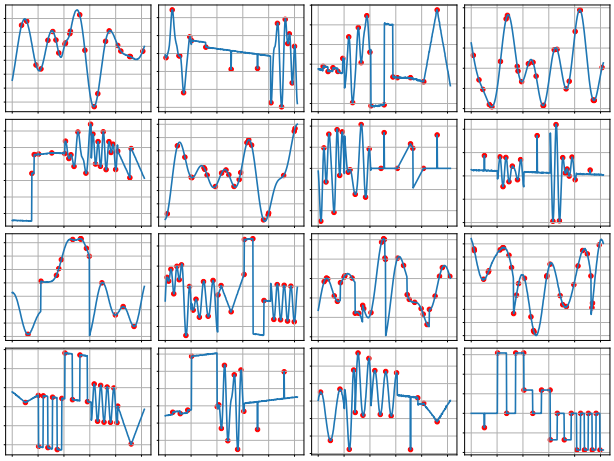


Figure 3. Samples from the synthetic dataset `SynthAlign`.

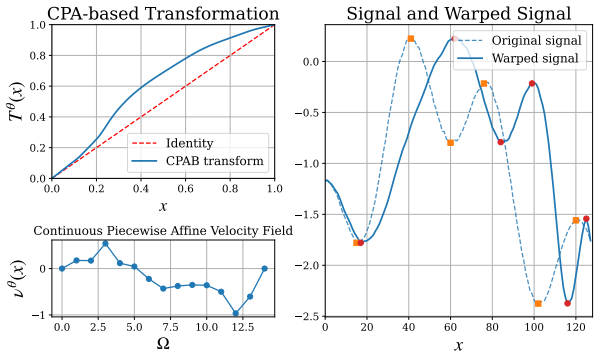


Figure 4. Generating signals and keypoints pairs with known correspondences using a CPAB transformation $T^{\boldsymbol{\theta}}$, which was obtained from a CPA velocity field, $\boldsymbol{v}^{\boldsymbol{\theta}}$, as proposed in (Freifeld et al., 2017) and briefly explained in our Appendix.

- **A 1D Keypoint Detection and Description Framework:** We introduce *TimePoint*, a self-supervised KP detection and description method for time series data.
- **A Synthetic Dataset for Time Series Alignment:** We design a synthetic time series dataset (`SynthAlign`) with known KPs and apply CPAB warps to generate training pairs with ground-truth correspondences.
- **Efficient Sparse Alignment via Multiscale Architectures:** We adapt the WTConv architecture for 1D signals to enable scalable keypoint and descriptor extraction. Combined with sparse DTW over learned features, this yields faster and more accurate alignment at significantly reduced computational cost.

2. Related Work

Time Series Alignment. DTW (Sakoe, 1971; Sakoe & Chiba, 1978) is a classical method for aligning sequences with phase or speed variations, but suffers from $\mathcal{O}(L^2)$ complexity and sensitivity to noise. Variants like *SoftDTW* (Cuturi & Blondel, 2017), *ShapeDTW* (Zhao & Itti, 2018), and *GI-DTW* (Vayer et al., 2020) improve differentiability or local structure modeling, but retain quadratic costs. *FastDTW* (Salvador & Chan, 2007) offers approximate linear-time alignment, though later shown to often be slower than exact DTW (Wu & Keogh, 2020). Unlike these methods, *TimePoint* uses sparse, learned KPs and descriptors to accelerate DTW while improving robustness.

CPAB Transformations. Modeling the nonlinear temporal distortions that time series often exhibit is a non-trivial task. Unlike image pairs, which can often be modeled via homo-

graphies, there is no gold standard transformation family for time series. *CPAB* transformations (Freifeld et al., 2017) provide a flexible and efficient way to model diffeomorphic time warping (Mumford & Desolneux, 2010). Prior works used them for weakly supervised averaging (Weber et al., 2019; Martinez et al., 2022), but not for learning KP-based alignment representations.

Deep Learning for Alignment. Recent works (Weber & Freifeld, 2023; Su & Wen, 2022; Xu et al., 2023; Zhang et al., 2023) learn alignment paths directly, often with high complexity or task-specific supervision. Related efforts in video (Trigeorgis et al., 2016; Cao et al., 2020; Li et al., 2022; Dwibedi et al., 2019) focus on few-shot action recognition but do not produce reusable descriptors or support zero-shot generalization.

TimePoint departs from prior work by learning temporal KPs and descriptors from synthetic data alone. It enables scalable, zero-shot alignment with sparse DTW, offering a foundation-style alternative to previous approaches.

3. SynthAlign: Synthetic Data Generation for Keypoint Learning

We first propose `SynthAlign`, a synthetic dataset generator for learning temporal keypoints and descriptors in a self-supervised setting (see Figure 2, left). It addresses key challenges in adapting 2D keypoint frameworks (e.g., SuperPoint (DeTone et al., 2018)) to 1D signals: (i) nonlinear temporal distortions not captured by low-dimensional transformations, and (ii) amplitude variations caused by noise or changes in sampling rates, can obscure salient events and complicate the task of identifying KPs.

3.1. Signal Generation

We synthesize pairs of time series with known keypoints and correspondences. Each signal $X \in \mathbb{R}^L$ (with KPs $Y \in \{0,1\}^L$) is generated by composing basic waveform:

- **Sine Wave Composition:** A combination of sine waves with varying frequencies and amplitudes to simulate oscillatory patterns.
- **Block, Triangle, and Sawtooth Waves:** Signals with square, triangular, or sawtooth waveforms to represent abrupt changes or linear ramps.
- **Radial Basis Functions (RBF):** Mixtures of Gaussian blobs to model localized smooth events.

KPs are derived from signal structure (e.g., peaks, endpoints, derivative zero crossings). To improve diversity, we apply augmentations such as Gaussian noise, linear trends, and flips. See Figure 3 for examples.

3.2. Generating Correspondences via CPAB Warps

To simulate realistic temporal distortions, we apply CPAB transformations (Freifeld et al., 2017), a parametric family of diffeomorphisms constructed from continuous piecewise affine (CPA) velocity fields. Given a signal X, we sample a transformation $T^{\boldsymbol{\theta}}$ and generate a warped version $X' = X \circ T^{\boldsymbol{\theta}}$, preserving known correspondences.

We sample $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\text{CPA}})$ from a smoothness prior (Freifeld et al., 2017), with $\sigma_{\text{smooth}} = 1$ and $\sigma_{\text{var}} = 0.5$, and use 16 CPA segments. This balances local expressiveness with realistic global structure. See Figure 4 and Appendix D for additional visualizations.

4. TimePoint Architecture

TimePoint is a self-supervised model that detects and describes KPs for alignment tasks. It consists of a shared encoder and two task-specific decoders (see Figure 2).

4.1. Architecture

Encoder. The input $X \in \mathbb{R}^L$ is processed by a fully convolutional encoder built with 3 levels of WTConv layers (Finder et al., 2024), enabling multiscale representation learning. Each block uses a stride of 2, yielding a downsampling factor of 8 and producing a feature map $F \in \mathbb{R}^{D_{\text{enc}} \times L'}$.

Keypoint Decoder. A conv layer maps F to $\mathbb{R}^{8 \times L'}$, which is reshaped to L and passed through a sigmoid to yield scores $S = (s_t)_{t=1}^L$. We apply NMS to select top-K KPs.

Descriptor Decoder. This decoder maps F into descriptor space ($D_{\text{desc}} = 256$), which is upsampled to L and ℓ_2-normalized, yielding dense descriptors $F_{\text{desc}} \in \mathbb{R}^{D_{\text{desc}} \times L}$. Sparse descriptors are extracted at predicted KPs locations.

4.2. Loss Functions

Detection Loss. We use binary cross-entropy over KP scores and labels from `SynthAlign`:

$$\mathcal{L}_{\text{kp}}(S, Y) = -\frac{1}{L} \sum_{t=1}^{L} [y_t \log(s_t) + (1 - y_t) \log(1 - s_t)]. \tag{1}$$

Descriptor Loss. We apply a contrastive loss using cosine similarity between corresponding KP descriptors from X and $X' = X \circ T^{\boldsymbol{\theta}}$:

$$\mathcal{L}_{\text{desc}}(D, D') = \frac{1}{N^2} \sum_{i,j=1}^{N} \Big[\mathbf{1}_{\mathcal{G}}(i,j) \max(0, m_p - \cos(D_i, D'_j))^2 + (1 - \mathbf{1}_{\mathcal{G}}(i,j)) \max(0, \cos(D_i, D'_j) - m_n)^2 \Big], \tag{2}$$

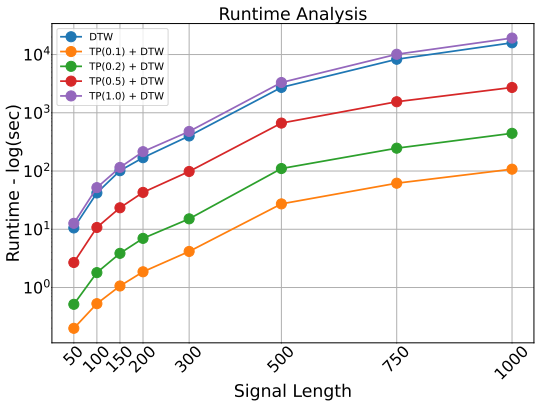


Figure 5. Runtime analysis. DTW KNN runtime between two synthetic datasets of $N = 500$ and varying lengths (on GPU).

where $m_p = 1$, $m_n = 0.1$. The combined loss is

$$\mathcal{L}(S, S', Y, Y', D, D') = \underbrace{\mathcal{L}_{\text{kp}}(S, Y)}_{\text{kp detection in } X} + \underbrace{\mathcal{L}_{\text{kp}}(S', Y')}_{\text{kp detection in } X'} + \underbrace{\mathcal{L}_{\text{desc}}(D, D')}_{\text{descriptor matching}}. \quad (3)$$

DTW with Sparse Descriptors At inference time, we extract descriptors D, D' from two inputs X, X'. DTW is computed on descriptors using a cosine-based cost:

$$\text{cost}(t, t') = 1 - \cos(D[t], D'[t']). \quad (4)$$

Critically, we can choose to compute DTW between D and D' *restricted to the KPs*. This drastically reduces the DTW complexity from $O(L \cdot L')$ to $O(\widetilde{L} \cdot \widetilde{L}')$ where $\widetilde{L} < L$ and $\widetilde{L}' < L'$ are numbers of KPs in X and X', respectively. For example, if the number of KPs for each signal is 10% of the length, this translates to a 100x speedup.

5. Experiments and Results

We evaluate **TimePoint** (TP) on real-world classification tasks and runtime efficiency using DTW-*k*NN. All results use TP trained solely on `SynthAlign`. Additional figures, results, and an ablation study are presented in the appendix.

5.1. Classification Performance and Runtime

We evaluate TP's *k*NN accuracy across 102 UCR datasets using a critical difference diagram (Demšar, 2006; Middlehurst et al., 2024), which ranks each classifier based on average rank and groups statistically indistinguishable methods using Wilcoxon signed-rank tests with Holm correction. As shown in Figure 6, *TP+DTW* achieves the highest average rank at both 100% and 20% KP usage, and remains competitive at 10%. We also compare TP+DTW and TP+SoftDTW to their full-length counterparts using batch-wise *k*NN on GPU (Table 1). With 20% of the KPs,

Table 1. Comparison of DTW and SoftDTW on 102 UCR datasets w/o TimePoint at various KP percentages. Top: 1-NN classification accuracy. Bottom: total runtime in GPU hours.

Method	**TimePoint 1-NN Accuracy**				
	Baseline	**10%**	**20%**	**50%**	**100%**
DTW	0.706	0.707	0.721	0.710	**0.732**
SoftDTW($\gamma = 0.1$)	0.677	0.659	0.662	0.689	**0.720**
SoftDTW($\gamma = 1$)	0.671	0.654	0.659	0.687	**0.711**
SoftDTW($\gamma = 10$)	0.670	0.655	0.658	0.680	**0.703**
Method	**GPU Runtime (hours)**				
	Baseline	**10%**	**20%**	**50%**	**100%**
DTW	192	0.71	2.88	19.59	193
SoftDTW($\gamma = 0.1$)	2.10	0.65	0.70	1.00	2.17
SoftDTW($\gamma = 1$)	1.98	0.52	0.56	0.86	2.16
SoftDTW($\gamma = 10$)	1.94	0.50	0.55	0.85	2.02

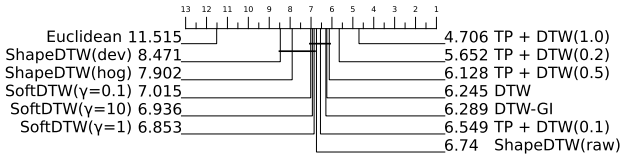


Figure 6. Critical Difference Diagram. The scores represent the average rank (1-NN Acc.) of each method across 102 UCR datasets.

TP+DTW improves accuracy by 2% and achieves a $\times 65$ **speedup** (3 vs. 192 hours). While SoftDTW performs best with higher KP ratios due to its smoothness bias, using 50% of the KPs still yields 1–2% higher accuracy at under half the runtime. These results highlight TP's ability to retain accuracy while drastically reducing computational cost.

5.2. Runtime Analysis

Runtime analysis on synthetic data (Figure 5) confirms that TP+DTW scales nearly linearly with sequence length, in contrast to standard DTW's quadratic behavior. At $L = 1000$, TP+DTW with 20% KPs is nearly two orders of magnitude faster than full-length DTW, highlighting TP's ability to maintain alignment quality while drastically reducing computational cost.

5.3. Conclusion

We introduced TimePoint (TP), a self-supervised framework for efficient time-series alignment. By leveraging synthetic data and employing CPAB transformations, TP learns to detect KPs and descriptors that enable sparse and accurate alignments. Our approach addresses the scalability limitation of traditional methods like DTW, achieving significant computational speedups and improved alignment accuracy.

Impact Statement This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Ansari, A. F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Arango, S. P., Kapoor, S., et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Cao, K., Ji, J., Cao, Z., Chang, C.-Y., and Niebles, J. C. Few-shot video classification via temporal alignment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10618–10627, 2020.

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

Dau, H. A., Bagnall, A., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., and Keogh, E. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

Demšar, J. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, 7:1–30, 2006.

DeTone, D., Malisiewicz, T., and Rabinovich, A. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 224–236, 2018.

Dwibedi, D., Aytar, Y., Tompson, J., Sermanet, P., and Zisserman, A. Temporal cycle-consistency learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1801–1810, 2019.

Finder, S. E., Amoyal, R., Treister, E., and Freifeld, O. Wavelet convolutions for large receptive fields. In *European Conference on Computer Vision*, pp. 363–380. Springer, 2024.

Freifeld, O., Hauberg, S., Batmanghelich, K., and Fisher III, J. W. Transformations based on continuous piecewise-affine velocity fields. *IEEE TPAMI*, 2017.

Fu, F., Chen, J., Zhang, J., Yang, C., Ma, L., and Yang, Y. Are synthetic time-series data really not as good as real data? *arXiv preprint arXiv:2402.00607*, 2024.

Li, S., Liu, H., Qian, R., Li, Y., See, J., Fei, M., Yu, X., and Lin, W. Ta2n: Two-stage action alignment network for few-shot action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 1404–1411, 2022.

Martinez, I., Viles, E., and Olaizola, I. G. Closed-form diffeomorphic transformations for time series alignment. In *International Conference on Machine Learning*, pp. 15122–15158. PMLR, 2022.

Middlehurst, M., Schäfer, P., and Bagnall, A. Bake off redux: a review and experimental evaluation of recent time series classification algorithms. *Data Mining and Knowledge Discovery*, pp. 1–74, 2024.

Mumford, D. and Desolneux, A. *Pattern theory: the stochastic analysis of real-world signals*. AK Peters/CRC Press, 2010.

Sakoe, H. Dynamic-programming approach to continuous speech recognition. *1971 Proc. the International Congress of Acoustics, Budapest*, 1971.

Sakoe, H. and Chiba, S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978. ISSN 0096-3518. doi: 10.1109/TASSP.1978.1163055.

Salvador, S. and Chan, P. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007.

Su, B. and Wen, J.-R. Temporal alignment prediction for supervised representation learning and few-shot sequence classification. In *International Conference on Learning Representations*, 2022.

Trigeorgis, G., Nicolaou, M. A., Zafeiriou, S., and Schuller, B. W. Deep canonical time warping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5110–5118, 2016.

Vayer, T., Chapel, L., Courty, N., Flamary, R., Soullard, Y., and Tavenard, R. Time series alignment with global invariances. *arXiv preprint arXiv:2002.03848*, 2020.

Weber, R. S. and Freifeld, O. Regularization-free diffeomorphic temporal alignment nets. In *International Conference on Machine Learning*, pp. 30794–30826. PMLR, 2023.

Weber, R. S., Eyal, M., Skafte Detlefsen, N., Shriki, O., and Freifeld, O. Diffeomorphic temporal alignment nets. In *Advances in neural information processing systems*, volume 32, 2019.

Wu, R. and Keogh, E. J. Fastdtw is approximate and generally slower than the algorithm it approximates. *IEEE Transactions on Knowledge and Data Engineering*, 34 (8):3779–3785, 2020.

Xu, M., Garg, S., Milford, M., and Gould, S. Deep declarative dynamic time warping for end-to-end learning of alignment paths. *arXiv preprint arXiv:2303.10778*, 2023.

Zhang, J., Zheng, S., Cao, W., Bian, J., and Li, J. Warpformer: A multi-scale modeling approach for irregular clinical time series. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3273–3285, 2023.

Zhao, J. and Itti, L. shapedtw: Shape dynamic time warping. *Pattern Recognition*, 74:171–184, 2018.

TimePoint: Appendix

Table of Contents

A. Ablation Study

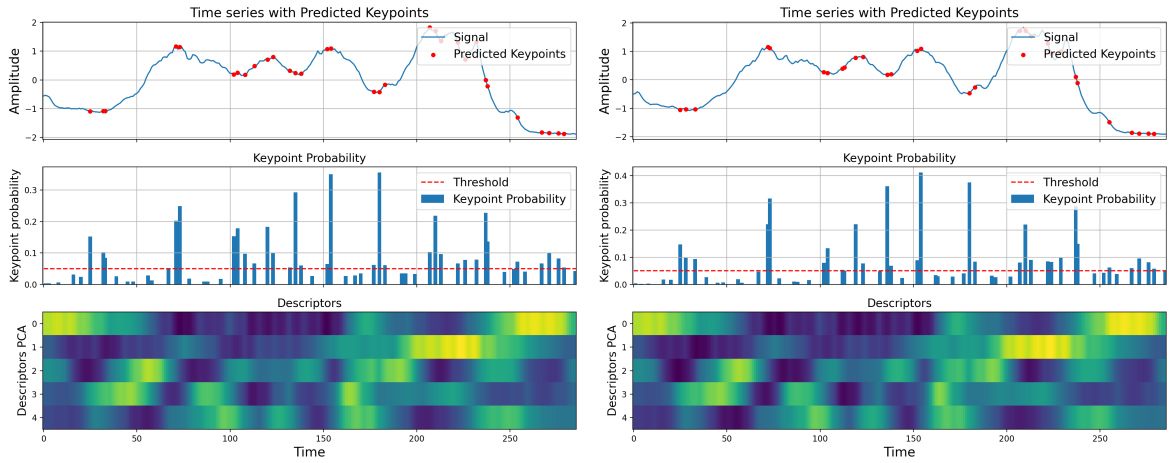
Table A.1 summarizes an ablation study over 27 datasets using a 1-NN accuracy metric. With the full sequence, the standard DTW (baseline) achieves an accuracy of 0.797. Performing DTW with TP features improves performance for both Dense Conv (∼400K parameters, 0.82) and WTConv (∼200K parameters, 0.815) encoders when using Euclidean distance. Switching to cosine similarity further boosts performance to 0.835 for Dense Conv and 0.869 for WTConv, highlighting the effectiveness of the wavelet-based encoder. Reducing the signal to just 20% of its length yields comparable trends: first, using TP merely as subsampling (i.e., ignoring the descriptors and using DTW on the raw signals restricted to *TimePoint*'s KPs) gives a 1-NN accuracy of 0.792, while switching to using TP's descriptors, with either the Dense Conv or WTConv encoders, reaches 0.826 or 0.865, respectively. This shows that *TimePoint* differs from naive subsampling methods by not only learning an input-dependent KP detection scheme tailored to alignment but also providing descriptors that, crucially, capture non-local context through a large receptive field. This leads to efficient DTW alignment without sacrificing accuracy. Note also TP+WTConv nearly matches in accuracy the full-length setting even though far fewer KPs are used. Lastly, dropping the NMS decreases TP+WTConv to 0.841.

Table A.1. Ablation Study

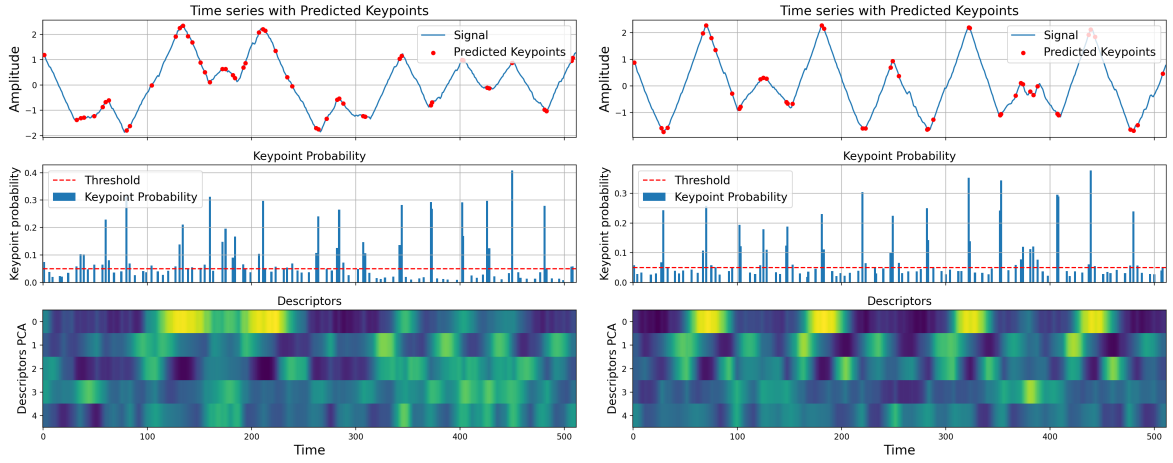
Encoder	Dist.	%L	NMS	1-NN acc.
DTW (baseline)	Euclidean	100%	-	0.797
TP + Dense Conv	Euclidean	100%	-	0.82
TP + WTConv	Euclidean	100%	-	0.815
TP + Dense Conv	Cosine Sim.	100%	-	0.835
TP + WTConv	Cosine Sim.	100%	-	**0.869**
TP + DTW (no descriptors)	Euclidean	20%	✓	0.792
TP + Dense Conv	Cosine Sim.	20%	✓	0.826
TP + WTConv	Cosine Sim.	20%	✗	0.841
TP + WTConv	Cosine Sim.	20%	✓	0.865

B. Additional Results

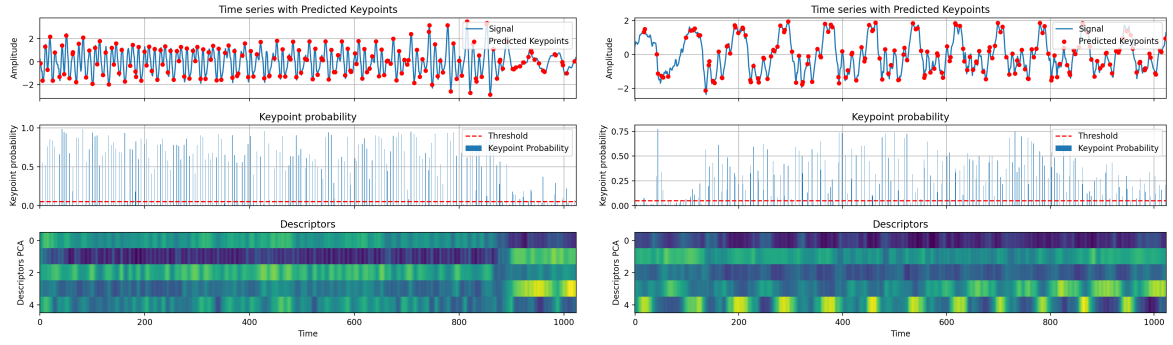
B.1. TimePoint Keypoints and Descriptors for Time Series of Different Lengths



(a) TimePoint keypoints and descriptors for the `Coffee` dataset ($L = 286$)



(b) TimePoint keypoints and descriptors for the `BeetleFly` dataset ($L = 512$)



(c) TimePoint keypoints and descriptors for the `Phoneme` dataset ($L = 1024$)

B.2. 1-NN Classification Comparisons

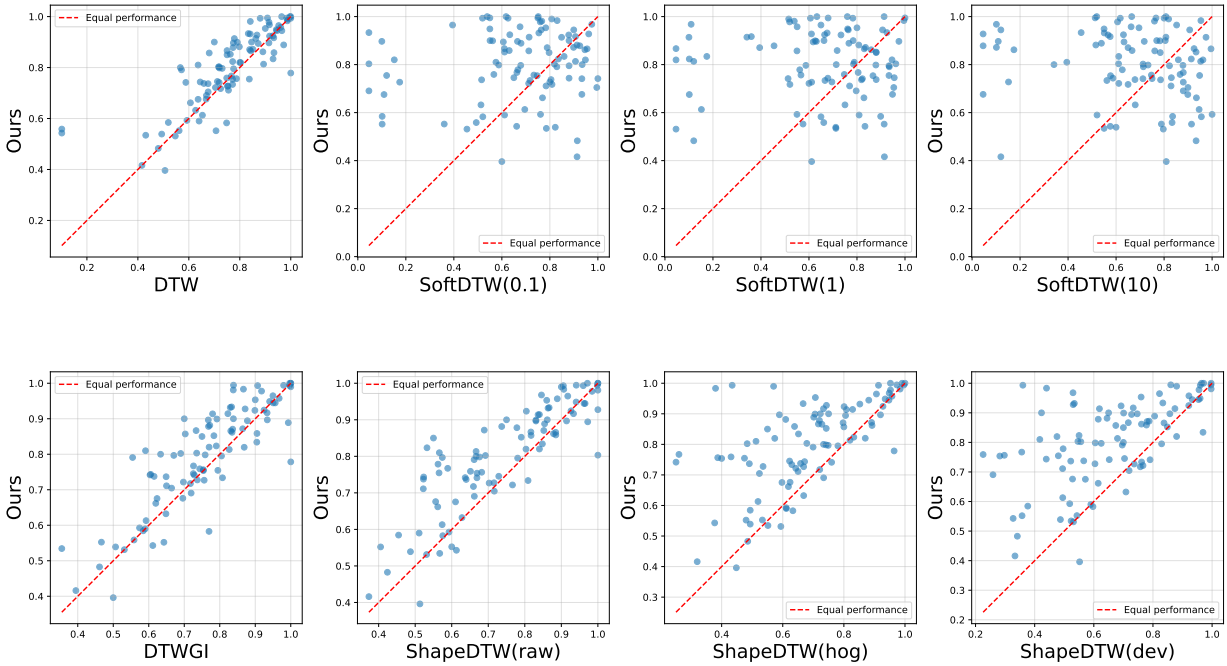


Figure B.2. **TP with 100 Percentage** of signal length vs. several DTW-based methods. Every dot represent a dataset

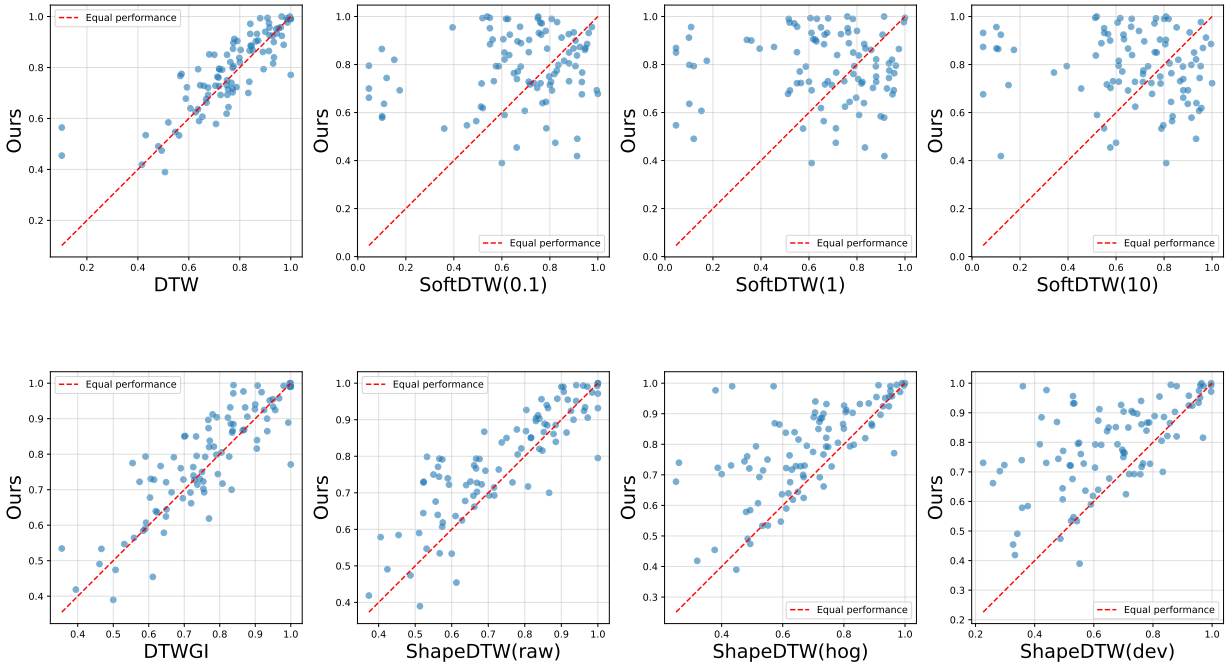


Figure B.3. **TP with 20 Percentage** of the keypoints vs. several DTW-based methods. Every dot represent a dataset

B.3. GPU RAM Consumption Analysis

Table B.1. **Total GPU memory usage comparison**. Summary of seven UCR datasets and the corresponding GPU memory usage (in GB, rounded to integers) for 1-NN DTW. The *TP + DTW* columns represent the memory footprint at different keypoint ratios, while standard DTW uses the full signal length.

Dataset				DTW	TP + DTW			
	Train	Test	Length		10%	20%	50%	100%
ChlorineConcentration	467	3840	166	186	2	8	47	186
Crop	7200	16800	46	996	17	46	260	996
ECG5000	500	4500	140	167	2	7	42	167
TwoPatterns	1000	4000	128	248	3	11	63	248
UWaveGestureLibraryX	896	3582	315	1194	13	49	302	1194
Wafer	1000	6164	152	538	6	22	136	538
Yoga	300	3000	426	611	7	25	154	611

Memory Consumption and Channel Independence. As shown in Table B.1, when both DTW and TP+DTW are applied to the full sequence (keypoint ratio $= 100\%$), the GPU memory consumption remains the same despite the fact that TP features are 256-dimensional. The key reason lies in the memory structure of our DTW implementation.

Dynamic Programming (DP) Matrix. The largest tensor created during DTW is a 4D DP matrix `D` of shape $\big[\texttt{batch\_size\_x},\ \texttt{batch\_size\_y},\ \texttt{length\_x} + 1,\ \texttt{length\_y} + 1\big]$. Notably, this matrix does *not* include any channel dimension. As a result, increasing the feature dimensionality from, say, 1 to 256 does not change the DP matrix size.

Cost Computation Over Channels. At each time step (i, j), we compute a scalar cost by reducing across the channel dimension, whether using cosine similarity or ℓ_2 distance. Consequently, the DP matrix stores only a single cost value for each pair (i, j), independent of the descriptor dimensionality.

Equal Memory Footprint at Full Length. When keypoint ratio $= 100\%$, TP+DTW and plain DTW both operate on the entire time series. Although *TimePoint* descriptors have a higher channel count (e.g., 256), that extra dimensionality is collapsed into a single scalar during the pairwise cost computation, leading to an identical memory footprint for the DP matrix. Hence, the total GPU RAM consumption scales only with the product of the sequence lengths and batch sizes, rather than the descriptor dimensionality or number of channels.

C. SynthAlign

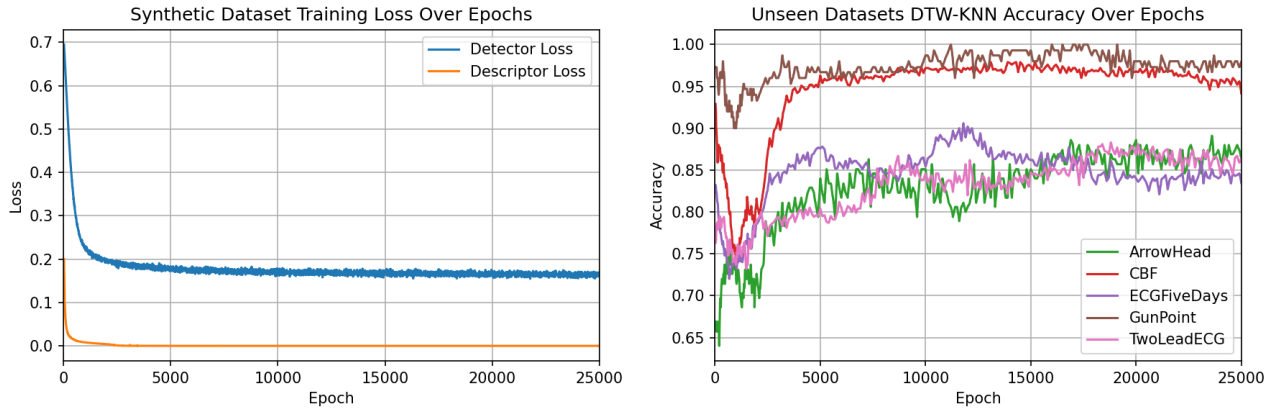


Figure C.1. Left: Training losses on `SynthAlign` data. Right: DTW-KNN accuracy on **unseen** datasets. As the training losses decrease the accuracy increases, indicating the loss functions, along with the synthetic training framework (`SynthAlign`) generalize well to real-world data for the task of identifying keypoints and learning descriptors. Epochs are trimmed to 25K for visibility purposes.

Does synthetic training generalize to real-world data? A notable finding is that *jointly minimizing the keypoint detection and descriptor losses on the synthetic dataset* leads to improved k-NN accuracy on *unseen* real-world data, as evident from Figure C.1. This suggests that learning to detect salient points and produce consistent descriptors under controlled, yet varied, synthetic data and distortions allows for alignment capabilities that transfer beyond the training distribution.

Data generation To train and evaluate our method under diverse temporal patterns and keypoint structures, we introduce the `SynthAlign` dataset. Each sample comprises a univariate time series of length L and an associated keypoint (KP) mask that marks salient events (e.g., peaks, start/end points). We generate data on-the-fly using a composition of procedurally defined waveforms and optional augmentations.

`SynthAlign` randomly draws from four principal waveform generators with specified probabilities:

1. *Sine Wave Composition*: Superposes multiple sine waves (random frequency, amplitude, and phase), with derivative-based KPs for local maxima or minima.
2. *Block Wave*: Creates square-wave-like segments with variable block sizes and amplitude, marking boundaries as KPs.
3. *Sawtooth Wave*: Forms a sawtooth signal of random frequency, designating signal resets as KPs.
4. *Radial Basis Function (RBF)*: Summation of Gaussian "blobs," entered at a random position; KPs appear at blob peaks.

Each generated signal has length $L = 512$ by default, though the code supports arbitrary lengths.

Composition and Augmentations. After selecting one or more waveform generators (drawn with probabilities $[0.6, 0.15, 0.05, 0.2]$ from the waveforms mentioned in the list above), the resulting signals are summed to form a final sample. We also introduce:

- **Linear Trends:** Randomly superimposed slopes and intercepts to simulate mild non-stationarity.
- **Flips:** Inverts a random subsection of the signal.
- **Noise:** Adds Gaussian noise sampled from $\mathcal{N}(0, 0.1)$ to further diversify training samples.

Keypoint Extraction. Keypoints originate from local maxima/minima (*sine* and *sawtooth*), block boundaries, Gaussian centers (*RBF*), and explicit markings for flips and linear boundaries. We ensure start/end points are included only when warranted by the signal design. This strategy provides a rich variety of salient events, allowing models to learn robust keypoint detection across diverse waveform shapes. Overall, `SynthAlign` delivers a flexible pipeline for generating synthetic time series with automatically labeled keypoints, serving as a valuable testbed for alignment, detection, and descriptor-learning techniques. Figure C.2 shows additional sample drawn from `SynthAlign`

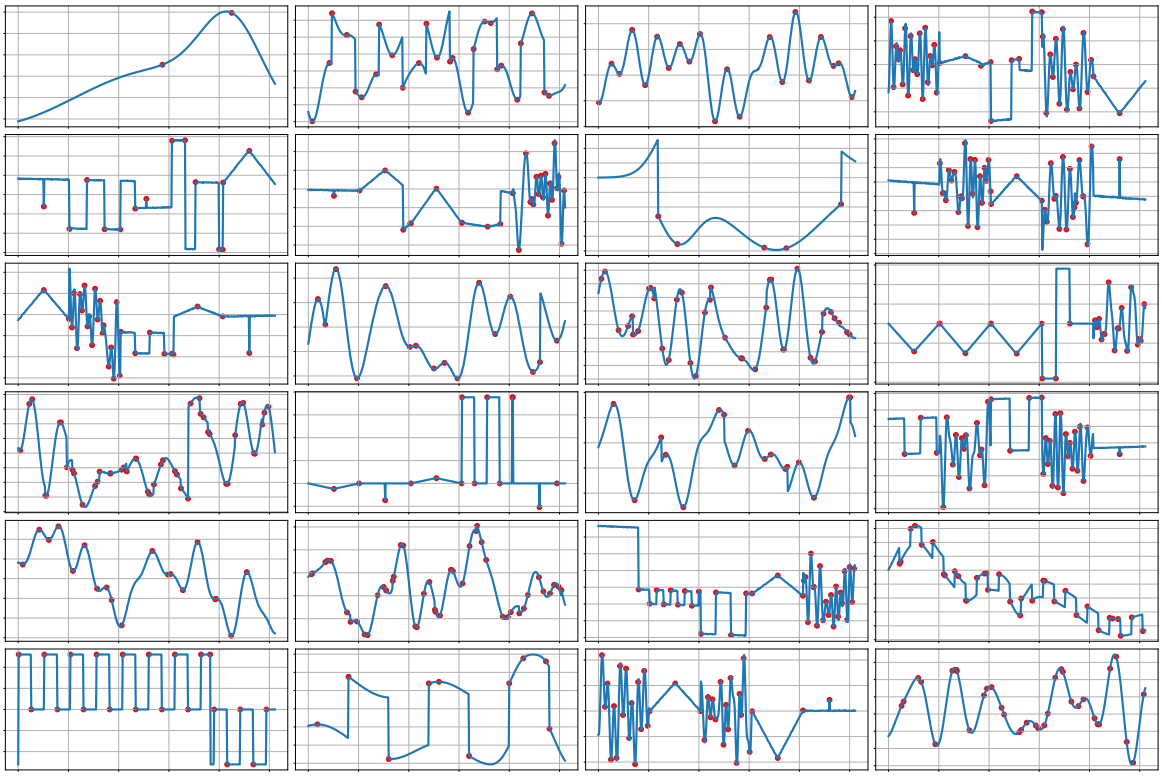


Figure C.2. More Samples from the `SynthAlign`.

D. CPAB Transformations

Let $T^{\boldsymbol{\theta}}$ be a diffeomorphism parameterized by $\boldsymbol{\theta}$. The main reason we chose CPAB transformations (Freifeld et al., 2017) as the parametric diffeomorphism family to be used within our method is that they are both expressive and efficient. Our presentation of CPAB transformations below closely follows (Freifeld et al., 2017), though we restrict the discussion to the 1D case (for the more general case, see (Freifeld et al., 2017)).

Let $\Omega = [a, b] \subset \mathbb{R}$ be a finite interval and let $\mathcal{V}$ be a space of continuous functions, from Ω to $\mathbb{R}$, that are also piecewise-affine w.r.t. some fixed partition of Ω into sub-intervals. Note that $\mathcal{V}$ is a finite-dimensional linear space. Let $d = \dim(\mathcal{V})$, let $\boldsymbol{\theta} \in \mathbb{R}^d$, and let $v^{\boldsymbol{\theta}} \in \mathcal{V}$ denote the generic element of $\mathcal{V}$, parameterized by $\boldsymbol{\theta}$. The space of CPAB transformations obtained via the integration of elements of $\mathcal{V}$, is defined as

$$\mathcal{T} \triangleq \left\{ T^{\boldsymbol{\theta}} : x \mapsto \phi^{\boldsymbol{\theta}}(x; 1) \text{ s.t. } \phi^{\boldsymbol{\theta}}(x; t) \text{ solves the integral equation } \phi^{\boldsymbol{\theta}}(x; t) = x + \int_0^t v^{\boldsymbol{\theta}}(\phi^{\boldsymbol{\theta}}(x; \tau))\, \mathrm{d}\tau \text{ where } v^{\boldsymbol{\theta}} \in \mathcal{V} \right\}. \tag{5}$$

Every $T^{\boldsymbol{\theta}} \in \mathcal{T}$ is an order-preserving transformation (i.e., it is monotonically increasing) and a diffeomorphism (Freifeld et al., 2017). Note that while $v^{\boldsymbol{\theta}} \in \mathcal{V}$ is CPA, the CPAB $T^{\boldsymbol{\theta}} \in \mathcal{T}$ is not (e.g., $T^{\boldsymbol{\theta}}$ is differentiable, unlike any non-trivial CPA function). Equation 5 also implies that the elements of $\mathcal{V}$ are viewed as velocity fields. Particularly useful for us are the following facts: 1) The finer the partition of Ω is, the more expressive the CPAB family becomes (which also means that d increases). 2) CPAB transformations lend themselves to a fast and accurate computation in closed form of $x \mapsto T^{\boldsymbol{\theta}}(x)$ (Freifeld et al., 2017). Together, these facts mean that *CPAB transformations provide us with a convenient and an efficient way to parameterize nonlinear monotonically-increasing functions*. Figure D.1 show random CPAB transformation applied to synthetic data sampled from `SynthAlign`, while Figure D.2 shows the effect of increasing the standard deviation (σ_{var}) when sampling $\boldsymbol{\theta}$ from the CPA smoothness prior.

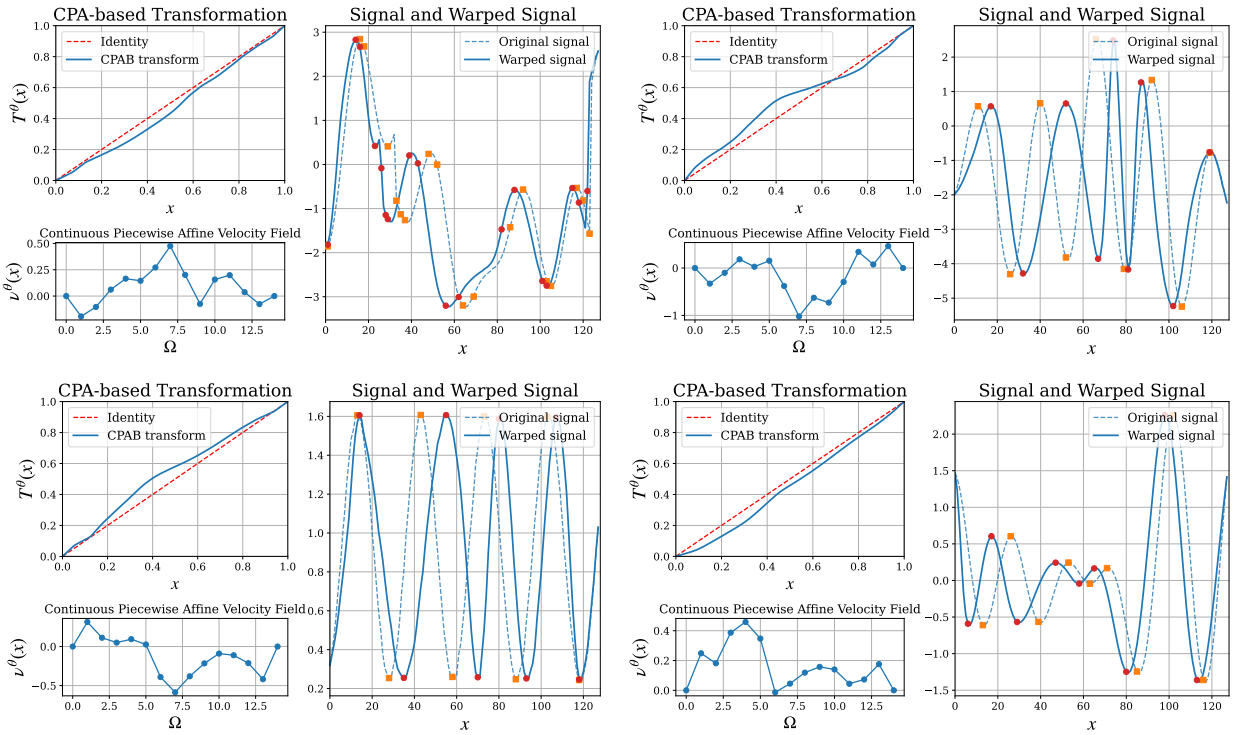


Figure D.1. Additional examples of the CPAB transformation.

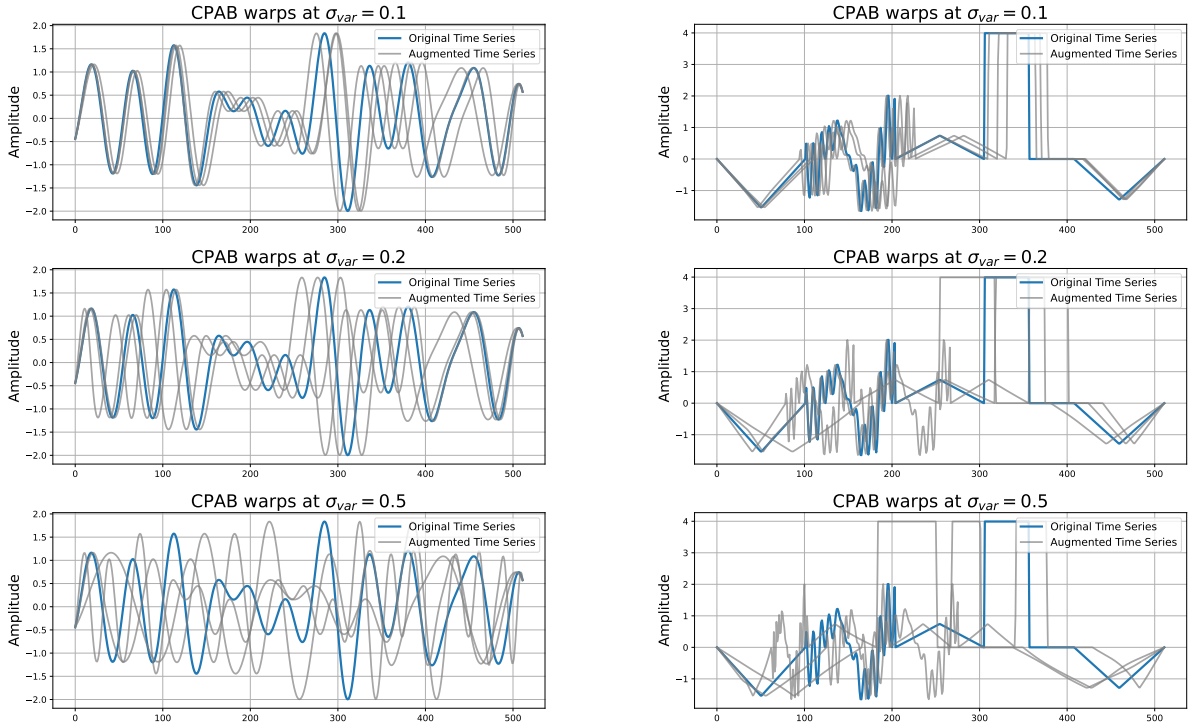


Figure D.2. `SynthAlign` synthetic data augmented with CPAB warps at increasing magnitude (top-to-bottom). Each panel shows the original signal in blue and 3 random augmentations in gray.

E. TimePoint Architecture Details

Table E.1. SuperPoint1D model architecture. Conv($C_{in} \to C_{out}$, k) denotes a 1D convolution from C_{in} to C_{out} channels with kernel size k, followed by BatchNorm and ReLU. WTConv denotes the wavelet-based convolution with group splitting and scale modules.

Stage	Layer	Details
		Encoder (WTConvEncoder1D)
layer1	ConvBlock1D	• Conv($1 \to 128$, $k = 3$, stride=1, padding=same) • BatchNorm(128), ReLU
layer2	WTConvBlock1D	• WTConv1d($128 \to 128$, kernel=3, groups=128) + scale modules • Conv($128 \to 128$, $k = 1$), BatchNorm(128), ReLU
layer3	WTConvBlock1D	• WTConv1d($128 \to 128$, kernel=3, groups=128) + scale modules • Conv($128 \to 256$, $k = 1$), BatchNorm(256), ReLU
layer4	WTConvBlock1D	• WTConv1d($256 \to 256$, kernel=3, groups=256) + scale modules • Conv($256 \to 256$, $k = 1$), BatchNorm(256), ReLU
		Detector Head (DetectorHead1D)
	Conv($256 \to 8$, $k = 1$) Sigmoid	Outputs detector logits for keypoint heatmap. Normalize the logits to probabilities
		Descriptor Head (DescriptorHead1D)
	Conv($256 \to 256$, $k = 1$) Upsample	Descriptor feature map. Upsample with scale factor = 8, mode=linear.

F. Full UCR Archive Results

F.1. UCR Archive Details

ID	Type	Name	Train	Test	Class	Length
1	Image	Adiac	390	391	37	176
2	Image	ArrowHead	36	175	3	251
3	Spectro	Beef	30	30	5	470
4	Image	BeetleFly	20	20	2	512
5	Image	BirdChicken	20	20	2	512
6	Sensor	Car	60	60	4	577
7	Simulated	CBF	30	900	3	128
8	Sensor	ChlorineConcentration	467	3840	3	166
9	Sensor	CinCECGTorso	40	1380	4	1639
10	Spectro	Coffee	28	28	2	286
11	Device	Computers	250	250	2	720
12	Motion	CricketX	390	390	12	300
13	Motion	CricketY	390	390	12	300
14	Motion	CricketZ	390	390	12	300
15	Image	DiatomSizeReduction	16	306	4	345
16	Image	DistalPhalanxOutlineAgeGroup	400	139	3	80
17	Image	DistalPhalanxOutlineCorrect	600	276	2	80
18	Image	DistalPhalanxTW	400	139	6	80
19	Sensor	Earthquakes	322	139	2	512
20	ECG	ECG200	100	100	2	96
21	ECG	ECG5000	500	4500	5	140
22	ECG	ECGFiveDays	23	861	2	136
23	Device	ElectricDevices	8926	7711	7	96
24	Image	FaceAll	560	1690	14	131
25	Image	FaceFour	24	88	4	350
26	Image	FacesUCR	200	2050	14	131
27	Image	FiftyWords	450	455	50	270
28	Image	Fish	175	175	7	463
29	Sensor	FordA	3601	1320	2	500
30	Sensor	FordB	3636	810	2	500
31	Motion	GunPoint	50	150	2	150
32	Spectro	Ham	109	105	2	431
33	Image	HandOutlines	1000	370	2	2709
34	Motion	Haptics	155	308	5	1092
35	Image	Herring	64	64	2	512
36	Motion	InlineSkate	100	550	7	1882
37	Sensor	InsectWingbeatSound	220	1980	11	256
38	Sensor	ItalyPowerDemand	67	1029	2	24
39	Device	LargeKitchenAppliances	375	375	3	720
40	Sensor	Lightning2	60	61	2	637
41	Sensor	Lightning7	70	73	7	319
42	Simulated	Mallat	55	2345	8	1024
43	Spectro	Meat	60	60	3	448
44	Image	MedicalImages	381	760	10	99
45	Image	MiddlePhalanxOutlineAgeGroup	400	154	3	80
46	Image	MiddlePhalanxOutlineCorrect	600	291	2	80
47	Image	MiddlePhalanxTW	399	154	6	80
48	Sensor	MoteStrain	20	1252	2	84
49	ECG	NonInvasiveFetalECGThorax1	1800	1965	42	750
50	ECG	NonInvasiveFetalECGThorax2	1800	1965	42	750
51	Spectro	OliveOil	30	30	4	570
52	Image	OSULeaf	200	242	6	427
53	Image	PhalangesOutlinesCorrect	1800	858	2	80
54	Sensor	Phoneme	214	1896	39	1024
55	Sensor	Plane	105	105	7	144
56	Image	ProximalPhalanxOutlineAgeGroup	400	205	3	80
57	Image	ProximalPhalanxOutlineCorrect	600	291	2	80
58	Image	ProximalPhalanxTW	400	205	6	80
59	Device	RefrigerationDevices	375	375	3	720
60	Device	ScreenType	375	375	3	720
61	Simulated	ShapeletSim	20	180	2	500
62	Image	ShapesAll	600	600	60	512
63	Device	SmallKitchenAppliances	375	375	3	720
64	Sensor	SonyAIBORobotSurface1	20	601	2	70
65	Sensor	SonyAIBORobotSurface2	27	953	2	65
66	Sensor	StarLightCurves	1000	8236	3	1024
67	Spectro	Strawberry	613	370	2	235
68	Image	SwedishLeaf	500	625	15	128
69	Image	Symbols	25	995	6	398
70	Simulated	SyntheticControl	300	300	6	60
71	Motion	ToeSegmentation1	40	228	2	277
72	Motion	ToeSegmentation2	36	130	2	343
73	Sensor	Trace	100	100	4	275
74	ECG	TwoLeadECG	23	1139	2	82

ID	Type	Name	Train	Test	Class	Length
75	Simulated	TwoPatterns	1000	4000	4	128
76	Motion	UWaveGestureLibraryAll	896	3582	8	945
77	Motion	UWaveGestureLibraryX	896	3582	8	315
78	Motion	UWaveGestureLibraryY	896	3582	8	315
79	Motion	UWaveGestureLibraryZ	896	3582	8	315
80	Sensor	Wafer	1000	6164	2	152
81	Spectro	Wine	57	54	2	234
82	Image	WordSynonyms	267	638	25	270
83	Motion	Worms	181	77	5	900
84	Motion	WormsTwoClass	181	77	2	900
85	Image	Yoga	300	3000	2	426
86	Device	ACSF1	100	100	10	1460
87	Sensor	AllGestureWiimoteX	300	700	10	Vary
88	Sensor	AllGestureWiimoteY	300	700	10	Vary
89	Sensor	AllGestureWiimoteZ	300	700	10	Vary
90	Simulated	BME	30	150	3	128
91	Traffic	Chinatown	20	343	2	24
92	Image	Crop	7200	16800	24	46
93	Sensor	DodgerLoopDay	78	80	7	288
94	Sensor	DodgerLoopGame	20	138	2	288
95	Sensor	DodgerLoopWeekend	20	138	2	288
96	EOG	EOGHorizontalSignal	362	362	12	1250
97	EOG	EOGVerticalSignal	362	362	12	1250
98	Spectro	EthanolLevel	504	500	4	1751
99	Sensor	FreezerRegularTrain	150	2850	2	301
100	Sensor	FreezerSmallTrain	28	2850	2	301
101	HRM	Fungi	18	186	18	201
102	Trajectory	GestureMidAirD1	208	130	26	Vary
103	Trajectory	GestureMidAirD2	208	130	26	Vary
104	Trajectory	GestureMidAirD3	208	130	26	Vary
105	Sensor	GesturePebbleZ1	132	172	6	Vary
106	Sensor	GesturePebbleZ2	146	158	6	Vary
107	Motion	GunPointAgeSpan	135	316	2	150
108	Motion	GunPointMaleVersusFemale	135	316	2	150
109	Motion	GunPointOldVersusYoung	136	315	2	150
110	Device	HouseTwenty	40	119	2	2000
111	EPG	InsectEPGRegularTrain	62	249	3	601
112	EPG	InsectEPGSmallTrain	17	249	3	601
113	Traffic	MelbournePedestrian	1194	2439	10	24
114	Image	MixedShapesRegularTrain	500	2425	5	1024
115	Image	MixedShapesSmallTrain	100	2425	5	1024
116	Sensor	PickupGestureWiimoteZ	50	50	10	Vary
117	Hemodynamics	PigAirwayPressure	104	208	52	2000
118	Hemodynamics	PigArtPressure	104	208	52	2000
119	Hemodynamics	PigCVP	104	208	52	2000
120	Device	PLAID	537	537	11	Vary
121	Power	PowerCons	180	180	2	144
122	Spectrum	Rock	20	50	4	2844
123	Spectrum	SemgHandGenderCh2	300	600	2	1500
124	Spectrum	SemgHandMovementCh2	450	450	6	1500
125	Spectrum	SemgHandSubjectCh2	450	450	5	1500
126	Sensor	ShakeGestureWiimoteZ	50	50	10	Vary
127	Simulated	SmoothSubspace	150	150	3	15
128	Simulated	UMD	36	144	3	150

F.2. UCR Archive Details

Table F.2: Accuracy Comparison Between TimePoint Configurations

Dataset/Method	TP+DTW				TP+SoftDTW ($\gamma = 1$)			
	(0.1)	(0.2)	(0.5)	(1)	(0.1)	(0.2)	(0.5)	(1)
Adiac	0.645	0.678	0.734	0.742	0.519	0.445	0.598	0.737
AllGestureWiimoteX	0.55	0.454	0.552	0.542	0.461	0.421	0.454	0.437
ArrowHead	0.869	0.851	0.874	0.857	0.669	0.657	0.806	0.840
BME	0.960	0.940	0.893	0.953	0.953	0.893	0.860	0.947
Beef	0.733	0.767	0.700	0.800	0.600	0.500	0.700	0.833
BeetleFly	0.950	0.850	0.900	0.900	0.850	0.950	0.850	0.950
BirdChicken	0.900	0.750	0.800	0.850	0.900	0.750	0.950	0.750
CBF	0.940	0.990	0.940	0.993	0.930	0.970	0.988	0.996
Car	0.850	0.850	0.767	0.867	0.633	0.733	0.783	0.817
Chinatown	0.831	0.924	0.915	0.945	0.828	0.901	0.974	0.942
ChlorineConcentration	0.574	0.624	0.648	0.632	0.527	0.538	0.574	0.635
Coffee	1.000	1.000	1.000	1.000	0.750	0.714	0.893	1.000
Computers	0.668	0.676	0.664	0.676	0.628	0.664	0.620	0.552
CricketX	0.715	0.703	0.551	0.754	0.610	0.585	0.685	0.685
CricketY	0.664	0.731	0.582	0.759	0.590	0.628	0.705	0.715
CricketZ	0.726	0.723	0.590	0.756	0.654	0.592	0.713	0.708
Crop	0.665	0.693	0.682	0.705	0.658	0.670	0.693	0.705
DiatomSizeReduction	0.967	0.958	0.967	0.958	0.850	0.794	0.905	0.958
DistalPhalanxOutlineAgeGroup	0.583	0.619	0.633	0.583	0.583	0.604	0.612	0.547
DistalPhalanxOutlineCorrect	0.652	0.692	0.699	0.717	0.616	0.685	0.678	0.714
DistalPhalanxTW	0.511	0.590	0.583	0.590	0.554	0.532	0.540	0.576
ECG200	0.850	0.820	0.820	0.820	0.820	0.820	0.900	0.800
ECG5000	0.921	0.924	0.921	0.924	0.922	0.916	0.922	0.924
ECGFiveDays	0.812	0.897	0.942	0.897	0.659	0.683	0.676	0.895
Earthquakes	0.669	0.662	0.691	0.691	0.676	0.662	0.619	0.676
ElectricDevices	0.593	0.607	0.616	0.613	0.588	0.592	0.602	0.615
FaceAll	0.676	0.717	0.720	0.734	0.634	0.667	0.691	0.730
FaceFour	0.807	0.932	0.886	0.932	0.693	0.727	0.932	0.943
FacesUCR	0.735	0.840	0.802	0.859	0.682	0.762	0.831	0.852
FiftyWords	0.745	0.760	0.736	0.802	0.684	0.668	0.677	0.754
Fish	0.874	0.903	0.880	0.914	0.731	0.709	0.834	0.891
FordA	0.780	0.775	0.768	0.791	0.717	0.729	0.754	0.782
FordB	0.641	0.640	0.647	0.662	0.586	0.633	0.640	0.623
FreezerRegularTrain	0.913	0.927	0.876	0.924	0.891	0.876	0.981	0.920
FreezerSmallTrain	0.780	0.793	0.760	0.798	0.758	0.783	0.862	0.771
Fungi	0.957	0.995	0.925	0.995	0.839	0.731	0.957	0.984
GestureMidAirD1	0.038	0.038	0.038	0.038	0.038	0.038	0.038	0.038
GestureMidAirD2	0.038	0.038	0.038	0.038	0.038	0.038	0.038	0.038
GestureMidAirD3	0.038	0.038	0.038	0.038	0.038	0.038	0.038	0.038
GesturePebbleZ1	0.163	0.163	0.163	0.163	0.163	0.163	0.163	0.163
GesturePebbleZ2	0.152	0.152	0.152	0.152	0.152	0.152	0.152	0.152
GunPoint	0.973	0.993	0.973	0.993	0.913	0.860	0.980	0.993
GunPointAgeSpan	0.981	0.975	0.972	0.978	0.953	0.924	0.972	0.965
GunPointMaleVersusFemale	0.994	1.000	0.997	1.000	0.978	0.994	1.000	1.000
GunPointOldVersusYoung	0.968	0.971	0.959	0.981	0.895	0.940	0.971	0.971
Ham	0.514	0.533	0.457	0.552	0.562	0.524	0.600	0.533
Herring	0.469	0.547	0.531	0.531	0.594	0.625	0.453	0.516
InsectEPGRegularTrain	0.964	0.932	0.876	0.928	0.855	0.827	0.807	0.819
AllGestureWiimoteY	0.558	0.564	0.578	0.558	0.492	0.47	0.47	0.457
AllGestureWiimoteZ	0.282	0.237	0.287	0.28	0.212	0.227	0.177	0.172
InsectEPGSmallTrain	0.815	0.795	0.703	0.803	0.622	0.755	0.711	0.747
InsectWingbeatSound	0.468	0.534	0.504	0.534	0.427	0.487	0.554	0.563
ItalyPowerDemand	0.941	0.934	0.957	0.945	0.943	0.939	0.944	0.949
LargeKitchenAppliances	0.757	0.744	0.709	0.755	0.640	0.613	0.589	0.560
Lightning2	0.836	0.869	0.852	0.820	0.738	0.803	0.820	0.852
Lightning7	0.671	0.740	0.808	0.767	0.616	0.658	0.658	0.740
Mallat	0.912	0.865	0.807	0.897	0.664	0.630	0.782	0.900
Meat	0.900	0.900	0.883	0.917	0.783	0.800	0.800	0.883
MedicalImages	0.687	0.714	0.729	0.728	0.674	0.672	0.722	0.714
MiddlePhalanxOutlineAgeGroup	0.422	0.390	0.461	0.396	0.364	0.429	0.396	0.442
MiddlePhalanxOutlineCorrect	0.663	0.729	0.694	0.722	0.625	0.601	0.622	0.722
MiddlePhalanxTW	0.442	0.474	0.539	0.539	0.422	0.487	0.526	0.578
MixedShapesSmallTrain	0.884	0.912	0.889	0.914	0.798	0.874	0.831	0.836
MoteStrain	0.826	0.861	0.852	0.862	0.807	0.863	0.865	0.856
NonInvasiveFetalECGThorax1	0.814	0.798	0.769	0.823	0.583	0.593	0.751	0.814
NonInvasiveFetalECGThorax2	0.847	0.867	0.846	0.872	0.693	0.677	0.825	0.860
OSULeaf	0.789	0.793	0.769	0.810	0.657	0.686	0.661	0.715
OliveOil	0.567	0.700	0.900	0.933	0.633	0.400	0.600	0.633
PhalangesOutlinesCorrect	0.690	0.727	0.748	0.741	0.671	0.697	0.681	0.754
Phoneme	0.267	0.268	0.261	0.284	0.212	0.203	0.166	0.165
PickupGestureWiimoteZ	0.100	0.100	0.100	0.100	0.100	0.100	0.100	0.100
Plane	1.000	0.990	1.000	1.000	1.000	0.981	1.000	0.990
PowerCons	0.856	0.906	0.878	0.894	0.811	0.867	0.911	0.861
ProximalPhalanxOutlineAgeGroup	0.790	0.805	0.805	0.795	0.810	0.780	0.800	0.800
ProximalPhalanxOutlineCorrect	0.818	0.821	0.825	0.852	0.832	0.821	0.818	0.845
ProximalPhalanxTW	0.673	0.693	0.712	0.727	0.683	0.727	0.751	0.712
RefrigerationDevices	0.525	0.491	0.504	0.483	0.461	0.451	0.429	0.483
ScreenType	0.413	0.419	0.411	0.416	0.376	0.397	0.387	0.384
ShakeGestureWiimoteZ	0.100	0.100	0.100	0.100	0.100	0.100	0.100	0.100
ShapeletSim	0.600	0.644	0.617	0.711	0.533	0.578	0.550	0.644
ShapesAll	0.860	0.873	0.862	0.878	0.728	0.782	0.777	0.803
SmallKitchenAppliances	0.555	0.579	0.600	0.552	0.557	0.576	0.507	0.475
SmoothSubspace	0.800	0.793	0.807	0.813	0.793	0.807	0.807	0.793
SonyAIBORobotSurface1	0.839	0.764	0.760	0.745	0.844	0.800	0.785	0.737
SonyAIBORobotSurface2	0.860	0.886	0.890	0.866	0.848	0.871	0.855	0.860
Strawberry	0.932	0.951	0.943	0.949	0.870	0.889	0.927	0.951

Dataset/Method	TP+DTW				TP+SoftDTW ($\gamma = 1$)			
	(0.1)	(0.2)	(0.5)	(1)	(0.1)	(0.2)	(0.5)	(1)
SwedishLeaf	0.875	0.904	0.888	0.899	0.790	0.811	0.866	0.904
Symbols	0.949	0.955	0.939	0.965	0.897	0.912	0.860	0.930
SyntheticControl	0.957	0.977	0.967	0.983	0.967	0.967	0.970	0.980
ToeSegmentation1	0.846	0.838	0.855	0.882	0.789	0.785	0.789	0.829
ToeSegmentation2	0.823	0.885	0.831	0.900	0.777	0.800	0.862	0.869
Trace	0.990	0.990	0.990	0.990	1.000	0.980	0.920	0.990
TwoLeadECG	0.830	0.816	0.801	0.834	0.789	0.695	0.785	0.837
TwoPatterns	0.729	0.771	0.687	0.778	0.681	0.676	0.602	0.756
UMD	0.910	0.889	0.840	0.889	0.854	0.819	0.812	0.854
UWaveGestureLibraryAll	0.917	0.956	0.927	0.968	0.784	0.913	0.910	0.905
UWaveGestureLibraryX	0.758	0.791	0.786	0.797	0.715	0.753	0.740	0.788
UWaveGestureLibraryY	0.703	0.730	0.716	0.743	0.643	0.682	0.660	0.711
UWaveGestureLibraryZ	0.689	0.728	0.701	0.738	0.652	0.693	0.688	0.714
Wafer	0.994	0.993	0.989	0.994	0.990	0.990	0.995	0.994
Wine	0.611	0.722	0.741	0.593	0.481	0.611	0.574	0.630
WordSynonyms	0.688	0.721	0.694	0.737	0.594	0.605	0.589	0.694
Worms	0.662	0.584	0.610	0.584	0.649	0.519	0.558	0.597
WormsTwoClass	0.727	0.636	0.649	0.675	0.688	0.649	0.623	0.688
Yoga	0.858	0.866	0.853	0.871	0.768	0.760	0.832	0.856

Table F.3: Accuracy Comparison Between Competitors.

Dataset/Method	DTW	DTW-GI	Euc.	ShapeDTW			SoftDTW		
				(dev)	(hog)	(raw)	($\gamma = 0.1$)	($\gamma = 1$)	($\gamma = 10$)
Adiac	0.588	0.604	0.066	0.652	0.251	0.637	1.000	0.513	0.750
AllGestureWiimoteX	0.135	0.611	0.101	0.327	0.377	0.613	0.662	0.833	0.576
ArrowHead	0.680	0.703	0.229	0.674	0.800	0.817	0.712	0.550	0.761
BME	0.900	0.900	0.493	0.760	0.707	0.860	0.783	0.800	0.696
Beef	0.567	0.633	0.267	0.700	0.733	0.667	0.935	0.714	0.341
BeetleFly	0.700	0.700	0.450	0.650	0.750	0.750	0.880	0.784	0.769
BirdChicken	0.750	0.750	0.800	0.700	0.550	0.550	0.797	0.880	0.784
CBF	1.000	0.997	0.658	0.360	0.434	0.906	0.626	0.783	0.800
Car	0.750	0.733	0.600	0.717	0.717	0.817	0.610	0.046	0.925
Chinatown	0.965	0.956	0.805	0.933	0.948	0.962	0.764	0.907	0.120
ChlorineConcentration	0.627	0.648	0.231	0.709	0.668	0.628	0.513	0.750	0.680
Coffee	0.964	1.000	0.536	0.964	1.000	1.000	0.752	0.913	0.808
Computers	0.668	0.696	0.672	0.528	0.624	0.556	0.933	0.948	0.046
CricketX	0.772	0.754	0.097	0.282	0.400	0.669	0.747	0.867	0.577
CricketY	0.749	0.744	0.074	0.226	0.431	0.651	0.567	0.747	0.867
CricketZ	0.787	0.754	0.092	0.297	0.387	0.682	0.733	0.567	0.747
Crop	0.676	0.664	0.052	0.719	0.524	0.716	0.995	0.953	0.899
DiatomSizeReduction	0.961	0.967	0.431	0.922	0.958	0.889	0.676	0.551	0.663
DistalPhalanxOutlineAgeGroup	0.748	0.770	0.604	0.597	0.633	0.576	0.521	0.650	0.955
DistalPhalanxOutlineCorrect	0.725	0.717	0.478	0.757	0.721	0.659	0.809	0.521	0.650
DistalPhalanxTW	0.640	0.590	0.468	0.590	0.612	0.511	0.611	0.809	0.521
ECG200	0.800	0.770	0.350	0.880	0.870	0.840	0.152	0.946	0.859
ECG5000	0.930	0.925	0.137	0.921	0.928	0.926	0.633	0.665	0.631
ECGFiveDays	0.775	0.768	0.551	0.747	0.920	0.830	0.665	0.631	0.717
Earthquakes	0.669	0.719	0.698	0.259	0.734	0.662	0.046	0.797	0.880
ElectricDevices	0.653	0.592	0.232	0.495	0.519	0.574	0.696	0.152	0.946
FaceAll	0.772	0.808	0.019	0.762	0.630	0.809	0.717	0.667	0.562
FaceFour	0.841	0.830	0.364	0.534	0.852	0.864	0.754	0.676	0.551
FacesUCR	0.934	0.905	0.143	0.778	0.639	0.885	0.631	0.717	0.667
FiftyWords	0.716	0.690	0.022	0.556	0.484	0.692	0.819	0.785	0.712
Fish	0.863	0.823	0.257	0.874	0.829	0.840	0.714	0.341	0.808
FordA	0.571	0.555	0.484	0.696	0.699	0.661	0.829	0.935	0.714
FordB	0.606	0.620	0.505	0.615	0.619	0.562	0.769	0.829	0.935
FreezerRegularTrain	0.917	0.899	0.546	0.692	0.801	0.804	0.879	0.733	0.567
FreezerSmallTrain	0.720	0.759	0.703	0.605	0.742	0.676	0.587	0.879	0.733
Fungi	0.909	0.839	0.038	0.860	0.973	0.941	0.550	1.000	0.513
GestureMidAirD1	0.046	0.538	0.046	0.485	0.400	0.508	0.684	0.754	0.676
GestureMidAirD2	0.046	0.438	0.046	0.338	0.431	0.454	0.739	0.684	0.754
GestureMidAirD3	0.046	0.169	0.046	0.346	0.300	0.292	0.576	0.739	0.684
GesturePebbleZ1	0.174	0.616	0.174	0.174	0.581	0.750	0.808	0.907	0.360
GesturePebbleZ2	0.152	0.563	0.152	0.203	0.563	0.722	0.341	0.808	0.907
GunPoint	0.880	0.907	0.513	0.960	0.913	0.960	0.519	0.606	0.618
GunPointAgeSpan	0.915	0.918	0.690	0.956	0.953	0.984	0.760	0.519	0.606
GunPointMaleVersusFemale	0.997	0.997	0.867	0.997	0.991	1.000	0.539	0.760	0.519
GunPointOldVersusYoung	0.841	0.838	0.514	0.997	0.984	1.000	0.766	0.539	0.760
Ham	0.562	0.467	0.486	0.543	0.533	0.600	0.360	0.575	0.789
Herring	0.547	0.531	0.406	0.531	0.594	0.531	0.455	0.046	0.797
InsectEPGRegularTrain	0.867	0.871	0.703	0.530	0.743	1.000	0.962	0.610	0.046
AllGestureWiimoteY	0.154	0.558	0.1	Nan	Nan	Nan	0.493	0.661	0.833
AllGestureWiimoteZ	0.094	0.288	0.11	Nan	Nan	Nan	0.493	0.661	0.852
InsectEPGSmallTrain	0.719	0.735	0.691	0.546	0.679	1.000	0.046	0.962	0.610
InsectWingbeatSound	0.431	0.355	0.091	0.523	0.552	0.567	0.785	0.712	0.550
ItalyPowerDemand	0.946	0.950	0.532	0.954	0.881	0.965	0.899	0.764	0.907
LargeKitchenAppliances	0.837	0.795	0.355	0.480	0.475	0.565	0.120	0.933	0.948
Lightning2	0.803	0.869	0.541	0.475	0.574	0.803	0.948	0.046	0.962
Lightning7	0.767	0.726	0.151	0.356	0.260	0.589	0.663	0.805	0.880
Mallat	0.914	0.934	0.244	0.857	0.589	0.914	0.100	0.975	0.109
Meat	0.933	0.933	0.333	0.733	0.733	0.933	0.907	0.360	0.575
MedicalImages	0.754	0.737	0.478	0.604	0.536	0.716	0.800	0.696	0.152
MiddlePhalanxOutlineAgeGroup	0.506	0.500	0.208	0.552	0.448	0.513	0.600	0.611	0.809

TimePoint

Dataset/Method	DTW	DTW-GI	Euc.	ShapeDTW			SoftDTW		
				(dev)	(hog)	(raw)	($\gamma = 0.1$)	($\gamma = 1$)	($\gamma = 10$)
MiddlePhalanxOutlineCorrect	0.704	0.698	0.584	0.766	0.643	0.766	0.712	0.600	0.611
MiddlePhalanxTW	0.494	0.506	0.448	0.487	0.494	0.487	0.823	0.712	0.600
MixedShapesSmallTrain	0.779	0.780	0.223	0.619	0.808	0.836	0.849	0.100	0.975
MoteStrain	0.891	0.835	0.570	0.761	0.892	0.879	0.946	0.859	0.174
NonInvasiveFetalECGThorax1	0.772	0.790	0.024	0.550	0.833	0.532	0.835	0.101	0.913
NonInvasiveFetalECGThorax2	0.851	0.864	0.032	0.787	0.891	0.689	0.899	0.835	0.101
OSULeaf	0.636	0.591	0.198	0.417	0.512	0.566	0.575	0.789	0.395
OliveOil	0.833	0.833	0.133	0.833	0.667	0.867	0.046	0.925	0.455
PhalangesOutlinesCorrect	0.719	0.728	0.503	0.790	0.760	0.669	0.933	0.823	0.712
Phoneme	0.272	0.228	0.011	0.082	0.047	0.117	0.952	0.849	0.100
PickupGestureWiimoteZ	0.120	0.220	0.120	0.280	0.480	0.700	0.833	0.576	0.739
Plane	1.000	1.000	0.095	0.971	0.952	0.971	0.618	0.633	0.665
PowerCons	0.872	0.878	0.506	0.728	0.806	0.972	0.606	0.618	0.633
ProximalPhalanxOutlineAgeGroup	0.776	0.805	0.820	0.829	0.780	0.780	0.880	0.933	0.823
ProximalPhalanxOutlineCorrect	0.763	0.784	0.550	0.849	0.742	0.790	0.611	0.880	0.933
ProximalPhalanxTW	0.751	0.756	0.737	0.737	0.668	0.702	0.174	0.611	0.880
RefrigerationDevices	0.480	0.461	0.384	0.341	0.485	0.424	0.914	0.120	0.933
ScreenType	0.416	0.395	0.400	0.333	0.320	0.373	0.913	0.914	0.120
ShakeGestureWiimoteZ	0.120	0.400	0.120	0.500	0.680	0.700	0.650	0.852	0.493
ShapeletSim	0.756	0.650	0.506	0.494	0.650	0.522	0.784	0.769	0.829
ShapesAll	0.773	0.768	0.058	0.615	0.720	0.778	0.925	0.455	0.046
SmallKitchenAppliances	0.707	0.643	0.336	0.357	0.480	0.405	0.101	0.913	0.914
SmoothSubspace	0.893	0.827	0.740	0.680	0.820	0.667	0.907	0.120	0.952
SonyAIBORobotSurface1	0.712	0.725	0.696	0.704	0.622	0.729	0.650	0.955	0.830
SonyAIBORobotSurface2	0.843	0.831	0.643	0.837	0.728	0.885	0.955	0.830	0.995
Strawberry	0.943	0.941	0.643	0.959	0.935	0.941	0.550	0.761	0.550
SwedishLeaf	0.790	0.792	0.088	0.701	0.706	0.830	0.562	0.626	0.783
Symbols	0.953	0.950	0.370	0.822	0.907	0.918	0.395	0.650	0.852
SyntheticControl	0.987	0.867	0.203	0.440	0.380	0.907	0.830	0.995	0.953
ToeSegmentation1	0.798	0.772	0.649	0.645	0.610	0.737	0.913	0.808	0.516
ToeSegmentation2	0.846	0.838	0.838	0.423	0.723	0.862	0.551	0.663	0.805
Trace	0.990	1.000	0.190	0.880	0.570	0.900	0.808	0.516	0.819
TwoLeadECG	0.931	0.904	0.500	0.969	0.651	0.848	0.859	0.174	0.611
TwoPatterns	1.000	1.000	0.255	0.496	0.964	0.562	0.667	0.562	0.626
UMD	0.972	0.993	0.792	0.806	0.701	0.854	0.680	0.766	0.539
UWaveGestureLibraryAll	0.916	0.891	0.138	0.529	0.948	0.845	0.974	0.780	0.928
UWaveGestureLibraryX	0.731	0.671	0.162	0.625	0.749	0.574	0.679	0.587	0.879
UWaveGestureLibraryY	0.645	0.606	0.149	0.439	0.671	0.523	0.880	0.679	0.587
UWaveGestureLibraryZ	0.659	0.615	0.129	0.566	0.658	0.523	0.805	0.880	0.679
Wafer	0.984	0.980	0.834	0.996	0.996	0.999	0.750	0.680	0.766
Wine	0.593	0.574	0.500	0.519	0.611	0.593	0.761	0.550	1.000
WordSynonyms	0.676	0.649	0.045	0.522	0.491	0.639	0.516	0.819	0.785
Worms	0.519	0.584	0.416	0.377	0.494	0.455	0.101	0.899	0.835
WormsTwoClass	0.636	0.623	0.429	0.571	0.597	0.610	0.109	0.101	0.899
Yoga	0.839	0.836	0.455	0.780	0.797	0.852	0.789	0.395	0.650