Hierarchical Clustering using Auto-encoded Compact Representation for Time-series Analysis

Soma Bandyopadhyay∗, Anish Datta and Arpan Pal
TCS Research, TATA Consultancy Services, Kolkata, India
{soma.bandyopadhyay, anish.datta, arpan.pal}@tcs.com

Abstract
Getting a robust time-series clustering with best choice of distance measure and appropriate representation is always a challenge. We propose a novel mechanism to identify the clusters combining learned compact representation of time-series, Auto Encoded Compact Sequence (AECS) and hierarchical clustering approach. Proposed algorithm aims to address the large computing time issue of hierarchical clustering as learned latent representation AECS has a length much less than the original length of time-series and at the same time want to enhance its performance. Our algorithm exploits Recurrent Neural Network (RNN) based under complete Sequence to Sequence(seq2seq) auto-encoder and agglomerative hierarchical clustering with a choice of best distance measure to recommend the best clustering. Our scheme selects the best distance measure and corresponding clustering for both univariate and multivariate time-series. We have experimented with real-world time-series from UCR and UCI archive taken from diverse application domains like healthcare, smart-city, manufacturing etc. and CricketX (human motion) etc. for smart-city. We have demonstrated our experimental analysis using both univariate and multivariate time-series from UCR Time Series Classification Archive[Bagnall et al., 2018b] and UCI Machine Learning Repository[Dua and Graff, 2017].

Our prime contributions are as follows:
1. **AECS and clustering on compact representation**: Auto encoded compact representation using Seq2Seq auto-encoder (AE)[Sutskever et al., 2014] to generate the latent representation of time-series. It has under complete architecture, and the learned latent representation has a length much less than the original length of time-series. Agglomerative hierarchical clustering is applied on AECS.
2. **Application of appropriate distance measure**: Chebyshev(CH), Manhattan(MA) and Mahalanobis(ML) are used as distance measures. These distance measures except ML take very low computing time to form the clusters.
3. **Choice of best distance measure**: We use Modified Hubert statistic(T) [Hubert and Arabie, 1985] as an internal clustering validation measure. It computes the disagreement between every pair of time-series and evaluates separation between the clusters. We select the best clustering which has the highest T. Here usage of Mahalanobis(ML) distance to evaluate T is our unique contribution.
4. **Extensive analysis on diverse time-series**: We study diverse and wide time-series from UCR Time Series Classification Archive [Bagnall et al., 2018b] and UCI Machine Learning Repository[Dua and Graff, 2017] spread across various application domains. 52 (41 univariate and 11 multivariate) time-series are considered for experiment and analysis of our proposed method. We use Rand Index (RI) [Rand, 1971] as external validation measure to compare our algorithm with other benchmark algorithms. We demonstrate our proposed method is not only comparable to the established State-of-the-Art clustering algorithms like K-shape [Paparrizos and Gravano, 2015], and hierarchical clustering with parametric derivative dynamic time warping (HC-DDTW)[Łuczak, 2016] but outperforms them significantly.

Our algorithm is applicable both on univariate as well as

∗Contact Author

1 Introduction
Time-series learning quite often faces paucity of ground truth. Hence finding patterns, groups and sub groups become a challenge. Specifically dearth of label and cost of domain expert’s knowledge to analyse time-series in healthcare, manufacturing, smart-city trigger the need of unsupervised learning approaches inevitable.

In this work we propose a novel mechanism to identify the clusters combining learned compact representation of time-series along with hierarchical clustering approach [Friedman et al., 2001]. The ability of Hierarchical clustering to form non-convex clusters without requiring the number of clusters to be formed paves its way for many real world tasks. Our
multivariate time-series whereas the benchmark clustering methods are applicable only univariate time-series. In case of multivariate time-series we have compared the benchmarks from [Bagnall et al., 2018a] and MLSTM-FCN [Karim et al., 2019]. Additionally we show that our mechanism is computationally efficient due to compact latent representation and choice of appropriate distance measure. This is very useful for real-world time-series having large sequence length.

2 Related Works
Hierarchical clustering (HC) algorithms create a hierarchy of clusters either using an bottom-up (Agglomerative) or top-down approach (Divisive) [Kaufman and Rousseeuw, 2009]. In agglomerative approach, two of the existing clusters are merged recursively while in divisive method, existing clusters are split into two. Hierarchical clustering does not require the prior information of number of clusters to be formed. It makes it ideally suitable for a number of real world problems where number of clusters to be formed is not available. Additionally its ability to form any arbitrary shaped and sized clusters makes it useful for many diverse applications. But the high computation cost of Hierarchical clustering acts as its major disadvantage. Maciej Luczak proposed an algorithm [Luczak, 2016] based on agglomerative hierarchical clustering of time-series with a parametric Dynamic Time Warping (DTW) distance measure showing HC outperforming K-Means. Though this approach addresses the challenge of finding a suitable distance measure, its drawback is longer computing time as finding DTW [Sakoe and Chiba, 1978] between two time-series is extremely computation heavy. Online Divisive-Agglomerative Clustering (ODAC) [Rodrigues et al., 2006] is a well known algorithm for hierarchical clustering of time-series which uses a top-down (Divisive) approach. Initially it assumes all the input time-series as a single cluster and then starts splitting it in different clusters in subsequent iterations. The splitting is continued until a certain heuristic condition is satisfied.

Among partitional based clustering, K-means [MacQueen and others, 1967] algorithm is most popular one. K-Shape [Paparrizos and Gravano, 2015] is another partitional clustering algorithm applied vastly on time-series. It uses shape-based distance metric (SBD) to find similarity between two time-series. Using this SBD, centroids are computed using a shape-extraction method for each cluster and then assign points to the clusters whose centroids are nearest to them. Although partitional clustering algorithms takes low time, they suffer from a potential shortcoming of generating hyperspherical clusters which may not fit in most of real world datasets.

3 Proposed method
In this work we propose a novel method to perform clustering by learning Auto Encoded Compact Sequence (AECS), a latent representation and applying hierarchical clustering on AECS. Our method encompasses exploiting distance measures like ML (Mahalanobis) [De Maesschalck et al., 2000] and low computing measures like CH (Chebyshev), MA (Manhattan) [Wang et al., 2013] to perform agglomerative hierarchical clustering and selecting best choice of clusters applying Modified Hubert Statistic (T) as internal clustering measure. The general functional block of our proposed approach is illustrated in Figure 1.

3.1 AECS: Auto-encoded Compact Sequence
In our method we first learn a compact representation of the time-series by using an Seq2Seq LSTM [Hochreiter and Schmidhuber, 1997] auto-encoder. Let $X = \{X_1, X_2, ..., X_M\}$ be a set of $M$ time-series, where $X_i = \{x_{i,1}, x_{i,2}, ..., x_{i,k}, ..., x_{i,n}\}$. $x_{i,k}$ gives the value of $i^{th}$ time-series’s $k^{th}$ timestep, and $n$ is the total number of timesteps or length of the time-series. In case of univariate, $x_i$ is scalar and is vector in case of multivariate time-series. We learn compact representation of time-series using multi layer Seq2Seq LSTM auto-encoder. The encoder block of the auto-encoder consists of an input layer $h_0$ having same number of units as number of timesteps and two hidden layers $h_1$ and $h_2$ having lengths $h_{11}$ and $h_{12}$ respectively. We propose Auto-Encoded Compact Sequence $AECS \in R^{M \times h_{12}}$, which is the latent representation of layer $h_2$ of encoder. The length of $AECS$ is $h_{12}$ where $h_{12} < h_{11} < n$. A schematic illustration of the auto-encoder model is presented in Figure 2.

Figure 1: Functional Block of HC-AECS

Figure 2: Schematic illustration of AECS using multilayer auto-encoder

We aim to learn a compact representation having a length much less than the original time-series to reduce the computing time as well as to capture important characteristics of time-series. AECS is an undercomplete representation hence bound to capture the important features of the input time-series.

Encoder:
$$H_i = f(X_i) \leftarrow h_{i,k}, s_{i,k} \leftarrow f_{LSTM}(h_{i,k-1}, s_{i,k-1}; x_{i,k});$$
Decoder: 
\[ X'_i = g(H_i) \leftarrow h'_i,k, s'_i,k \leftarrow f_{LSTM}(h'_i,k-1, s'_i,k-1, h_i,k) \]

\[ i = 1, \ldots, T \]

Here \( l_{enc} = \text{length}(AECS) = h_{l2} \). The reconstruction loss is Mean Square Error (mse):
\[ l_{\text{rec}} = \frac{1}{Mn} \sum_{i=1}^{M} \sum_{j=1}^{n} (x_{i,j} - x'_{i,j})^2 \]

where \( X'_i = \{x'_{i,1}, x'_{i,2}, \ldots, x'_{i,n}\} \) is the reconstructed output for \( i^{th} \) instance.

### 3.2 HC using diverse distance measure

Hierarchical clustering (HC) produces a hierarchy of clusters formed either by merging or dividing existing clusters recursively. The iterative process of agglomeration/division of the clusters can be visualised in form of a tree-like structure known as dendrogram. Each node of the tree/dendrogram contains one or more time-series \((X_i)\) representing a cluster. Here we use agglomerative hierarchical clustering which takes each input time-series as an individual cluster and then start successively merging pairs of clusters having highest similarity among them until a single cluster is formed. Hence it measures similarities among the pair of time-series to put them into same cluster and the dissimilarities between the pair of groups of time-series to take the decision of further fusing the clusters. We have used average linkage [Friedman et al., 2001] to obtain pairwise inter-cluster dissimilarities. In average linkage, distance \( a_d \) between clusters \( C_0 \) and \( C_1 \) is:
\[ a_d(C_0, C_1) = \frac{1}{|C_0| \times |C_1|} \sum_{x_i \in C_0} \sum_{x_j \in C_1} \text{dist}(X_i, X_j) \]

\(|C_i|\) denotes cardinality or number of members in cluster \( i \).

Average linkage aims to form compact and relatively far apart clusters than other linkage methods like single or complete by keeping average pairwise dissimilarity. Usage of linkage methods like single or complete are scope of future study.

In this work we have used following three different distance measures to find the dissimilarity between two time-series (say \( X_i \) and \( X_j \)), where \( X_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,k}, \ldots, x_{i,n}\} \) and \( X_j = \{x_{j,1}, x_{j,2}, \ldots, x_{j,k}, \ldots, x_{j,n}\} \), \( x_{i,k} \) and \( x_{j,k} \) indicates \( k^{th} \) timestep of \( i^{th} \) and \( j^{th} \) time-series respectively as well as to obtain average linkage among the clusters as follows.

**Chebyshev Distance:**
\[ \max_{k=1}^{T} |x_{i,k} - x_{j,k}| \] (3):
The maximum distance between two data points in any single dimension.

**Manhattan Distance:**
\[ \sum_{k=1}^{T} |x_{i,k} - x_{j,k}| \] (4):
This belongs to Minkowski family which computes distance travelled to get from one data point to the other if a grid-like path is followed.

**Mahalanobis Distance:**
\[ \sqrt{(X_i - X_j)^T \cdot C^{-1} \cdot (X_i - X_j)} \] (5):
Finds the distance between two data points in multidimensional space. Here \( X_i \) and \( X_j \) are two time-series and \( C \) is the co-variance matrix between \( X_i \) and \( X_j \).

### 3.3 Choice of best clustering and corresponding distance measure

We use Modified Hubert statistic \((T)\), an internal clustering measure which evaluates sum of distance between each pair of instances weighted by distance between the centers of clusters which they belong to. We rank clustering formed by different distance measures based on \( T \). Best clustering has highest \( T \). In case two or more distance measures produces the highest value of \( T \), we report all of them as the best measures. We have measured disagreements between pairs of time-series and separation between clusters using ML (Mahalanobis) to evaluate \( T \), which is one of our prime contributions.

\[ T = \frac{2}{n(n-1)} \sum_{X_i \in X} \sum_{X_j \in X} d(X_i, X_j) d(c_i, c_j) \] (6)

\[ d(X_i, X_j) = d_{ML}(X_i, X_j) \ ; d(c_i, c_j) = d_{ML}(c_i, c_j) \] (7)

where \( C_i \) represents \( i^{th} \) cluster and \( c_i \) is the center of cluster \( C_i \), \( d_{ML}(X_i, X_j) \) is Mahalanobis distance between time-series \( X_i \) and \( X_j \) and \( d_{ML}(c_i, c_j) \) is Mahalanobis distance between centers of clusters to which \( X_i \) and \( X_j \) belongs.

**Algorithm 1 HC-AECS:** Hierarchical clustering with auto-encoded compact latent representation

**Input:** Time-series: \( X \in R^{M \times n} \). Number of time-series: \( M \).
Sequence length: \( n \). Encoder hidden layers length: \( h_{l1}, h_{l2} \); \( h_{l2} < h_{l1} < n \). Epochs: \( e \). Batch size: \( b \). Learning rate: \( lr \).
Number of clusters: \( K \) (optional)

**Output:** Best clustering with corresponding distance measure

1. **procedure** AECS \((X, h_{l1}, h_{l2}, e, b, lr)\)
2. Create a multilayer auto-encoder using Eq. (1).
3. Get latent representation from layer \( h_{l2} \) of encoder.
4. **return** AECS \( \in R^{M \times h_{l2}} \)
5. **end procedure**
6. **procedure** CLUSTER_AECS \((AECS, D=\{CH, MA, ML\})\)
7. **for** distance metric \( d_i \) in \( D \) **do**
8. \( C \leftarrow HC(AECS, K, d_i, \text{linkage} = \text{average}) \)
9. \( \text{Clusters}[d_i] \leftarrow C \)
10. **end for**
11. **return** Clusters
12. **end procedure**
13. **procedure** BEST_CLUSTER \((Clusters, D=\{CH, MA, ML\})\)
14. \( \text{max}_T \leftarrow 0 \)
15. \( d_{\text{best}} \leftarrow null \)
16. **for** distance metric \( d_i \) in \( D \) **do**
17. \( C \leftarrow \text{Clusters}[d_i] \)
18. \( T \leftarrow \text{Modified_HubertStatistic}(C) \) (based on Eq. (6) and (7))
19. **if** \( \text{max}_T < T \) **then**
20. \( d_{\text{best}} \leftarrow \emptyset \)
21. \( \text{max}_T \leftarrow T \)
22. \( d_{\text{best}} \leftarrow d_i \)
23. **else if** \( \text{max}_T = T \) **then**
24. \( d_{\text{best}} \leftarrow d_{\text{best}} \cup d_i \)
25. **end if**
26. **end for**
27. **return** \( \{\text{Clusters}[d_{\text{best}}], d_{\text{best}}\}, \forall i \)
28. **end procedure**

### 3.4 Validation

We compute Rand Index (RI) [Rand, 1971], one of the external clustering validation measures to compare the perfor-
mance of our proposed mechanism with the benchmark results. RI is a cluster validation measure which considers all pairs of time-series and counts number of pairs assigned in same or different clusters formed by our algorithm to the true clusters based on the given labels. $RI = TP + TN$.

4 Experimental Analysis

4.1 Data Description

We use univariate and multivariate time-series from UCR Time Series Classification Archive [Bagnall et al., 2018b] to evaluate the performance of our proposed algorithm. Additionally we perform our analysis on 2 multivariate datasets from UCI Machine Learning Repository [Dua and Graff, 2017]. Description of univariate and multivariate time-series are presented in Tables 1 and 2 respectively. Each dataset has default train-test split. In our approach we merge the train and test sets and perform Z-normalisation on the merged data.

Table 1: Dataset Description of UCR univariate time-series

| Dataset                 | #Train | #Test | Length | #class |
|-------------------------|--------|-------|--------|--------|
| DistalPhalanxOAG        | 400    | 139   | 80     | 3      |
| DistalPhalanxOC         | 600    | 276   | 80     | 2      |
| DistalPhalanxTW         | 400    | 139   | 80     | 6      |
| MiddlePhalanxOAG        | 400    | 154   | 80     | 3      |
| MiddlePhalanxOC         | 600    | 291   | 80     | 2      |
| MiddlePhalanxTW         | 399    | 154   | 80     | 6      |
| ProximalPhalanxOAC      | 600    | 291   | 80     | 2      |
| ProximalPhalanxTW       | 400    | 205   | 80     | 6      |
| Beef                    | 30     | 30    | 470    | 5      |
| Earthquakes             | 322    | 139   | 512    | 2      |
| Coffee                  | 28     | 28    | 268    | 2      |
| Fish                    | 175    | 175   | 463    | 7      |
| Ham                     | 109    | 105   | 431    | 2      |
| Strawberry              | 613    | 370   | 235    | 2      |
| DiatomSizeReduction     | 16     | 306   | 345    | 4      |
| Wine                    | 57     | 54    | 234    | 2      |
| ChlorineConcentration   | 467    | 3840  | 166    | 3      |
| SyntheticControl        | 300    | 300   | 60     | 6      |
| TwoPatterns             | 1000   | 4000  | 128    | 4      |
| CricketX                | 390    | 390   | 300    | 12     |
| CricketY                | 390    | 390   | 300    | 12     |
| CricketZ                | 390    | 390   | 300    | 12     |
| ECG200                  | 100    | 100   | 96     | 2      |
| ECG5000                 | 500    | 4500  | 140    | 5      |
| ECGFiveDays             | 23     | 861   | 136    | 2      |
| TwoLeadECG              | 23     | 1139  | 82     | 2      |
| Lightning7              | 70     | 73    | 319    | 7      |
| Plane                   | 105    | 105   | 144    | 7      |
| SonyAIBORobotSurface1   | 20     | 601   | 70     | 2      |
| SonyAIBORobotSurface2   | 27     | 953   | 65     | 2      |
| ArrowHead               | 36     | 175   | 251    | 3      |
| Swedish Shay            | 500    | 625   | 128    | 15     |
| Yoga                    | 300    | 3000  | 426    | 2      |
| Car                     | 60     | 60    | 577    | 4      |
| GunPoint                | 50     | 150   | 150    | 2      |
| Adiac                   | 390    | 391   | 176    | 37     |
| ToeSegmentation1        | 40     | 228   | 277    | 2      |
| ToeSegmentation2        | 36     | 130   | 343    | 2      |
| Wafer                   | 1000   | 6164  | 152    | 2      |
| FordB                   | 3636   | 810   | 500    | 2      |
| BME                     | 30     | 150   | 128    | 3      |

Table 2: Dataset Description of UCR and UCI Multivariate time-series

| Archive | Dataset     | #Train | #Test | Length | #class | #dim |
|---------|-------------|--------|-------|--------|--------|------|
| AtrialFibrillation | 15  | 15 | 640 | 3 | 2 |
| ERing | 30 | 30 | 65 | 6 | 4 |
| FingerMovements | 316 | 100 | 50 | 2 | 28 |
| HandMD | 160 | 74 | 400 | 10 |
| Libras | 180 | 180 | 45 | 15 | 2 |
| NATOPS | 180 | 180 | 51 | 6 | 24 |
| SelfRegulationSCP2 | 200 | 180 | 1152 | 2 | 7 |
| StandWalkJump | 12 | 15 | 2500 | 3 | 4 | 4 |

UCI Wafer 298 896 104 198 2 6 GesturePhase 198 198 4 214 5 18

4.2 Latent representation

We first compute Auto-Encoded Compact Representation for time-series following algorithm HC-AECS.AECS() stated in proposed method. We have used Mean-Squared Error (MSE) as loss function and Stochastic Gradient Descent (SGD) [Robbins and Monro, 1951] as optimizer with learning rate $(lr)$ as 0.004 and momentum as 0. We use batch size 32.

Let us take a representative dataset Adiac to explain thoroughly. It is an univariate time-series with 781 instances $(M=781)$ where each time-series is of length 176 $(n=176)$ i.e $X \in \mathbb{R}^{781 \times 176}$. The input layer $h_0$ produces a representation of the same length as that of the number of time steps i.e $h_0 \in \mathbb{R}^{781 \times 176}$. This representation is passed to first hidden layer $h_1$ which scales down the length of each instance to $h_{11} = 16$ $(h_1 \in \mathbb{R}^{781 \times 16})$. This compressed sequence is fed to layer $h_2$ of length $h_{11} = 12$ where we get a further compressed representation for each $X_i$ of Adiac as $AECS_{Adiac} \in \mathbb{R}^{781 \times 12}$.

Also let us consider a multivariate dataset ERing having 60 instances $(M=60)$ of time-series length 65 $(n=65)$ with 4 dimensions i.e $X \in \mathbb{R}^{60 \times 65 \times 4}$. Each timestep consists of a vector of size 4 corresponding to the dimensions. Here after passing the dataset through the input layer $h_0$ we get a representation for each time-series of length equal to the number of timesteps. Correspondingly layers $h_1$ and $h_2$ produces representations of length $h_{11} = 12$ for each time-series respectively.

4.3 Study on diverse distance measure and choice of best clustering

We started our analysis using 7 different distance measures - Chebyshev, Cosine, Euclidean, Canberra, Manhattan, Mahalanobis and Cross-correlation. After extensive analysis, we have concluded that Chebyshev, Manhattan and Mahalanobis performs better than the other measures on raw time-series as well as on proposed compact representation of time-series - AECS. Figure 3 depicts RI measures of HC on raw time-series using the above mentioned seven distance measures on four representative datasets. We can see either Chebyshev (CH), Manhattan (MA), Mahalanobis (ML) performs best in each one of these datasets. Hence, we use Chebyshev, Manhattan and Mahalanobis as the distance measures in our algorithm.

Next we calculate $T$ based on Eq.6 and Eq.7 and HC-AECS.BestCluster() on the clusterings formed by above con-
included three distance measures, $T_{CH}$ for CH, $T_{MA}$ for MA and $T_{ML}$ for ML. Extending the example presented in section 4.2, for Adiac we obtained $T_{CH} = 1.38$, $T_{MA} = 1.56$, $T_{ML} = 1.90$. Hence we choose clusters formed by ML as it has highest $T$. While validation we see RI of clusters formed by ML is indeed the highest among the three RI measures (RI (CH) = 0.698, RI (MA) = 0.780, RI (ML) = 0.926).

We illustrate further in Figure 4 the plot of $T$ vs RI for the said 3 distance measures on 3 univariate (DistalPhalanxOAG (DPOAG), CricketX, and SwedishLeaf) and 3 multivariate (Libras, Handwriting and NATOPS) time-series. In this figure we can observe that for each time-series, $T$ and RI are linearly dependent signifying the clustering having the best $T$ also has the highest RI using either CH, MA, ML. As depicted in the figure for the univariate time-series CH produces the best clustering for DPOAG and ML produces best clustering for CricketX and SwedishLeaf. Whereas for multivariate time-series, MA performs best for NATOPS and ML performs best for Libras and Handwriting.

### 4.4 Results

We compare HC-AECS with important State-of-the-Art clustering mechanisms on time-series like HC with parametric Dynamic Time Warping (DTW)-derivative(DDTW) (HC-DDTW) [Łuczak, 2016] and K-shape[ Paparrizos and Gravano, 2015]. Authors of [Łuczak, 2016] have already established that they perform better than K-means hence we don’t include K-means in our analysis.

We have used RI as external clustering measure to perform unbiased comparison with the prior work as it has presented performance measure using RI. Further we have extended our analysis by using Normalized Mutual Information (NMI) [Strehl and Ghosh, 2002] as external clustering measure across a subset of considered 52 time-series. We have observed that NMI reflects also the same trend as obtained using RI as depicted in Table 3. As an example DistalPhalanxOAG, where NMI(HC-DDTW) = 0.414, NMI(K-Shape) = 0.252 and NMI(HC-AECS) = {0.458, 0.321, 0.017} for the distance measures CH, MA and ML respectively. Here HC-AECS using CH gives the best performance as obtained using RI.

#### Analysis on univariate time-series: Comparison w.r.t State-of-the-Art methods

Table 3 depicts comparison of clustering performance of HC-AECS using the proposed combination of distance measures CH, MA and ML with established State-of-the-Art clustering algorithms. Here, RI is used for comparative analysis. We also present the choice of best clustering based on $T$ and corresponding distance measure following our proposed method. Additionally we have also demonstrated the performance of HC applied on raw time-series (HC-L) using the proposed distance measures. We indicate highest RI corresponding to the best distance measure in italics for HC-L and HC-AECS. Best RI achieved among all algorithms has been marked in bold. In summary, 76% cases HC-AECS outperforms HC-DDTW the benchmark hierarchical clustering method and remains within the range of 0.001 to 0.281 with average variation of 0.065, and also 51% cases outperforms K-Shape and remains within the range of 0.005 to 0.229 with average variation 0.1.

We have also compared performance of our algorithm with existing benchmark classification results using 14 representative univariate time-series in Table 4. The benchmark results for a dataset are reported for the algorithm which performs best among the 38 algorithms compared in [Bagnall et al., 2017]. In this case we have applied HC-AECS on the given test data as benchmark accuracy are reported for test data only. Here, also HC-AECS generates very close classification benchmark results and outperforms multiple cases. In the cases where classification results are higher, HC-AECS only vary 0.12 in average (approximately 14%) from benchmark classification results. The cases where test data contains less number of samples like DistalPhalanxOAG (139 test samples) and Lightning7 (73 test samples), HC-AECS lags behind.

Figure 5 depicts two-dimensional representation of proposed AECS using multi-layer undercomplete Seq2Seq autoencoder. We use t-SNE[Maaten and Hinton, 2008] with parameters: perplexity =30, learning rate=200 and iterations=100 to obtain 2-D plot for visualization. Two representative time-series ECGFiveDays (univariate) and Wafer(UCI) (multivariate) are considered here. The clear separation of classes can be visualised using our learned compact representation.
using Euclidean distance (1NN-ED), dimension-independent results with benchmark classification algorithms like 1-NN out of 30 multivariate time-series from UCR and compare our

Analysis on multivariate time-series
We perform HC-AECS and measure RI on 9 representative out of 30 multivariate time-series from UCR and compare our results with benchmark classification algorithms like 1-NN using Euclidean distance (1NN-ED), dimension-independent

DTW (1NN-DTW) and dimension-dependent DTW (1NN-DTW_p) [Bagnall et al., 2018a; Bagnall et al., 2017] as presented in Table 5. We can see in 5 out of 9 datasets, our algorithms performs better than the benchmark algorithms. Additionally we compare our model with State-of-the-Art multivariate time-series classification algorithm MLSTM-FCN [Karim et al., 2019] using 2 datasets from UCI repository presented in Table 6. Our approach outperforms MLSTM-FCN by a significant margin for both the cases.

Computational time for HC-AECS
HC-AECS consumes much less computation time as compared to HC-DDTW and HC with raw time-series due to the much reduced length of the latent representation. HC-DDTW requires a lot of computation power as performing the much reduced length of the latent representation. HC-AECS consumes much less computation time as compared to HC-DDTW, HC-L and HC-AECS.
all of which exploits hierarchical clustering. Here per instance cluster time is computed for the algorithms which is total time taken divided by number of instances in the dataset. The time required by HC-AECS can be divided into three parts: (a) time required to learn AECS($t_{aecs}$), (b) time required for clustering($t_{c}$), (c) time taken to choose best distance metric($t_{v}$). Figure 6a depicts computing time difference between epoch 10 and epoch 30. From our results, we conclude the computation time required for HC-AECS is approximately 27 times less than HC-DDTW. The time difference becomes more apparent in comparatively larger datasets like Adiac where HC-AECS performs almost 47 times faster than HC-DDTW. At the same time, $t_{aecs}$ does not vary significantly with increase in epochs.

HC-AECS approach performs approximately 2.4 times faster than K-Shape in average considering the clustering time($t_{c}$). Furthermore in datasets like Coffee and Beef where number of timesteps is high and number of instances is relatively low, our approach performs significantly faster than K-Shape. For Coffee (Samples = 56, timesteps = 286) HC-AECS runs 16 times faster while for Beef (Samples = 60, timesteps = 470) its performs 18 times faster. As K-Shape is not applicable for multivariate time-series we perform the time comparison analysis only on univariate time-series.

### 5 Conclusion

In this paper, we have presented a robust hierarchical clustering approach HC-AECS using a combination of hierarchical clustering with multi-layer under complete Seq2Seq auto-encoder representation. We have exploited compact latent representation of both univariate and multivariate time-series by selecting an appropriate distance measure. Proposed reduced representation AECS (Auto-Encoded Compact Sequence) and proper application of distance measures CH, MA
and ML address high computational cost issue of hierarchical clustering along with capturing important representation.

To this end, we have performed extensive analysis considering different distance measures and proposed a novel mechanism to choose the distance measure corresponding to the best clusters. We have used an internal clustering performance measure, Modified Hubert statistic $T$ for best clustering selection.

Our experimental analysis on UCR time-series from different application domains indicates the robust performance of our proposed scheme which is completely unsupervised. It outperforms 76% w.r.t State-of-the-Art hierarchical clustering based time-series clustering method. Additionally our proposed method HC-AECS is applicable both on univariate and multivariate time-series. Presence of longer sequence length time-series are frequent in healthcare, machine and manufacturing and smart-city application domains, here our method perfectly suits by representing a compact sequence which is constant across diverse length of time-series.

References

[Bagnall et al., 2017] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

[Bagnall et al., 2018a] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

[Bagnall et al., 2018b] Anthony Bagnall, Jason Lines, William Vickers, and Eamonn Keogh. The uca & ucr time series classification repository. *URL http://www.timeseriesclassification.com*, 2018.

[De Maesschalck et al., 2000] Roy De Maesschalck, Delphine Jouan-Rimbaud, and Désiré L Massart. The mahalanobis distance. *Chemometrics and intelligent laboratory systems*, 50(1):1–18, 2000.

[Dua and Graff, 2017] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[Friedman et al., 2001] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Hubert and Arabie, 1985] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[Karim et al., 2019] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. Multivariate lstm-cnns for time series classification. *Neural Networks*, 116:237–245, 2019.

[Kaufman and Rousseeuw, 2009] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

[Łuczak, 2016] Maciej Łuczak. Hierarchical clustering of time series data with parametric derivative dynamic time warping. *Expert Systems with Applications*, 62:116–130, 2016.

[Maaten and Hinton, 2008] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[MacQueen and others, 1967] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[Paparrizos and Gravano, 2015] John Paparrizos and Luis Gravano. k-shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1855–1870, 2015.

[Rand, 1971] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[Robbins and Monro, 1951] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[Rodrigues et al., 2006] Pedro Pereira Rodrigues, Joao Gama, and Joao Pedro Pedroso. Odac: Hierarchical clustering of time series data streams. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 499–503. SIAM, 2006.

[Sakoe and Chiba, 1978] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[Strehl and Ghosh, 2002] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617, 2002.

[Sutskever et al., 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[Wang et al., 2013] Xiaoyue Wang, Abdullah Mueen, Hui Ding, Goce Trajcevski, Peter Scheuermann, and Eamonn Keogh. Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, 26(2):275–309, 2013.