Interpretable Time Series Classification with Dynamic Routing

Wei Zhao
Department of Computer Science and Engineering
Washington University in St. Louis

Abstract

Time Series Classification (TSC) is a classical task in the community of data mining and machine learning. The approaches for TSC have been rapidly developed in recent years. Existing approaches for TSC can be grouped into shape based methods, structure based methods, ensemble based methods and deep learning based methods. In general, ensemble based and deep learning based methods have outperformed the other two methods regarding accuracy. But it is hard to interpret deep learning model and ensemble based model. Some of the shape based and structure based methods have interpretability but without good performance and efficiency. In this project, we propose a method based on dynamic routing and logistic regression. Our method can achieve as good performance as ensemble based and deep learning based methods and still retain interpretability. We conduct experiments on UCR time series dataset to analyze the accuracy, efficiency and interpretability of our proposed method.

1 Introduction

Our daily live is surrounding with time series data, like weather readings, stock prices and industrial observations. With the rapid increasing of computing power, how to extract knowledge from time series data attracts more and more attention. Time series classification (TSC) is one of the important task in the community of time series research. TSC has critical applications in the fields of health care, finance and industry. A lot of approaches for time series classification have been proposed in recent years, these approaches can mainly be grouped into four categories: shape based methods, structure based methods, ensemble based methods and deep learning based methods.

Shaped based methods use the raw numeric time series data directly with distance metric like Euclidean distance (ED) or Dynamic Time Warping (DTW) distance [1]. These method employ a classifier such as 1-Nearest Neighbor on the distance metric to make the final decision. These methods can achieve good performance but expensive. Furthermore, it is hard to know which aspect of the time series leads to the final decision. The other methods, shapelet based methods, aim to search for shapelet [2] from the raw numeric time series. Shapelets are discriminative subsequences which can be used for classification. Shapelet based methods are interpretable but still expensive. Some other work [3] focus on how to find shapelet more efficiently.

Structure based methods are based on the symbol representation of the raw numeric data. Symbolic Aggregate Approximation (SAX) [4] is one of the popular symbol representations, which is a sequence of symbols presenting the raw numerical data. SAX can make algorithms efficient by reducing the dimensionality of the raw numeric data, without negatively affecting their accuracy. Based on SAX, the work SAX-VSM [5] aims to find distinct SAX-words with TF-IDF weights. SAX-VSM is interpretable but still expensive, since SAX-SVM uses an optimization process to find the best SAX transformation parameters. Symbolic Fourier Approximation (SFA) [6] is another symbol representation of the raw time series data. Bag-of-SFA-Symbols (BOSS) [7] proposes a

distance metric based on histograms of SFA words. This distance metric is combine with 1-NN for the final decision.

Ensemble based methods ensemble different classifiers together to make the final decision. Shapelet ensemble (SE) [8] combines shapelet transformations with a heterogeneous ensemble method. The flat collective of transform-based ensembles (flat-COTE) [9] is an ensemble of 35 different classifiers based on features from time and frequency domains. Despite state-of-the-art performance of ensemble methods, these methods suffer high complexity during training and testing process. Since these methods ensemble multiple classifiers together, they usually lack interpretability.

Recently, with the rapid development of deep learning, more and more deep learning based methods have been proposed for time series classification. In [10], a multi-scale CNN (MCNN) has been proposed for univariate time series classification. In [11], the author applied Multilayer Perceptrons (MLP), Fully Connected Network (FCN) and Residual Neural Network (ResNet) [12] on time series classification task and achieve state-of-the-art performance. Even deep learning based methods have good performance with efficient inference stage, they also lack interpretability.

It is worth mentioning that Capsule Network [13] has attracted a lot attention with its special architecture design and routing algorithm. A lot of work based on Capsule Network have led to impressive results in computer vision [13], text classification [14], etc. In this project, we propose a novel model for time series classification based on convolution, dynamic routing and logistic regression. In our framework, convolution is firstly used to extract local features from the raw time series data. Then, dynamic routing is applied to group the correlated features together. Finally, logistic regression is used to make the final decision. As shown in our experiments on UCR dataset [15], our approach can achieve competitive performance compared with state-of-the-art methods. Furthermore, we use case study to show the interpretability of our model.

2 Related Work

Time series classification have been studied for a long time. A lot of approaches have been proposed for TSC. Since our work focus on interpretable approaches for TSC, we review some interpretable methods below.

Among shape based methods, shapelet based method have attracted lots of attention since the discriminative subsequence, shapelet, is detected in these methods. Shapelets make these methods interpretable by providing the subsequences which are crucial for time series. The first shapelet based method is proposed by Ye et al [2]. This method is expensive since it searches for shapelet in the subsequence space. To make the shapelet-based method more efficient, some other works, like learned shapelets (LS) [16] and Fast shapelet (FS) [3], are proposed to accelerate the searching process.

Among structure based methods, SAX-VSM [5] is one method which can provide interpretability. SAX-VSM firstly use the training set to build a dictionary of SAX words, and then computes a TF-IDF weight vector for each class. Since SAX-VSM relies on the parameters for SAX transformation, SAX-VSM have to employ an optimization algorithm with cross-validation to search for the optimal parameters. The optimization process make this method extremely expensive. The author show the interpretability of SAX-VSM by mapping the SAX word with high TF-IDF weight to the original time series, the corresponding region in time series is an important subsequence for the time series.

Recently, with the rapid development of deep learning, some deep learning based methods [10, 11] have been proposed for TSC. Even these methods have shown excellent performance on UCR dataset [15], they are unable to interpret what make the final decision. Capsule network [13] is a new kind of neural network. Every activation is a vector in Capsule Network not like a scalar in traditional neural network. Every vector represents a specific type of entity, the length of the vector represents the probability that the entity exists and its orientation represents the instantiation parameters. In capsule layer, dynamic routing is used to transfer information from lower layer to the subsequent layer. After dynamic routing, the correlated capsules in the low layer will be grouped together to a capsule in the subsequent layer.

In this project, we propose a framework with dynamic routing and logistic regression (DRLR) for interpretable time series classification. Our model is based on convolution, dynamic routing and logistic regression. The first step can extract local features from raw time series data. Then,

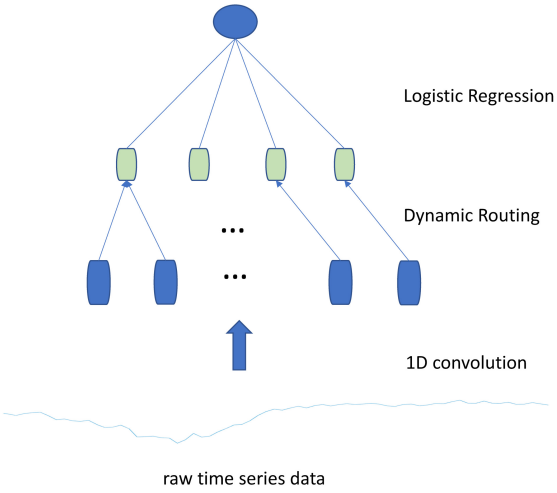

Figure 1: The architecture of DRLR

the correlated local features will be grouped together to constitute high level features. Logistic regression will make the final decision based on the high level features and it also provides the model interpretability since its weight representing the importance of corresponding features.

3 Method

The overall architecture of DRLR is shown in Figure 1. We describe our method by the following three sections.

3.1 Convolution

In order to extract local features from the raw time series data and prepare for the vector input for the next capsule layer, we use convolution to constitute a vector representation of local features. The input of the convolution layer is the raw time series data and the input channel is 1. After convolution, the output size will be (l, c), where l is the length, c is the number of output channels. These l vectors with c dimensions will be l input capsules for the next capsule layer. And each dimension of the c-D vector represents a specific feature of its corresponding subsequence in the raw time series.

3.2 Dynamic Routing

In capsule layer, dynamic routing algorithm is applied. The key idea of dynamic routing is to use an iterative manner to find a non-linear mapping which can send each capsule to an appropriate capsule in the subsequent layer. The capsule in the high layer represents high level features from a long subsequence in time series, and the capsule in the low layer represents low level feature from a short subsequent in time series. The high layer capsule is constituted by the low layer capsule which are sent to it. The algorithm procedure is shown in Algorithm 1.

First, the input vector u_i will be multiplied by a weight matrix W_{ij}, which encodes the relationship between lower level feature u_i and higher level feature s_j, to generate a vector $\hat{u}_{j|i}$.

$$\hat{u}_{j|i} = W_{ij} u_i \tag{1}$$

$\hat{u}_{j|i}$ is an estimation of the output s_j of the capsule j in the viewpoint of u_i. With the help of this estimation, the output s_j of capsule j is defined as a weighted sum over all these vectors $\hat{u}_{j|i}$. s_j is computed by

$$s_j = \sum_i c_{ij} \hat{u}_{j|i} \tag{2}$$

Algorithm 1 Dynamic Routing

procedure Routing($\hat{u}_{j|i}$, r, l)
for all capsule i in layer l and capsule j in layer $(l+1)$:$b_{ij} = 0$.
for r iterations **do**
  for all capsule i in layer l: $c_i = softmax(b_i)$
  if the last iteration **then**
    for all capsule i in layer l: $c_{ij} = mask(c_{ij})$
  end if
  for all capsule j in layer $(l+1)$: $s_j = \sum_i c_{ij}\hat{u}_{j|i}$
  for all capsule j in layer $(l+1)$: $v_j = squash(s_j)$
  for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} = b_{ij} + \hat{u}_{j|i} \cdot v_j$
end for
return v_j

where the c_{ij} are the routing coefficients that are determined by the iterative dynamic routing process.

Next, a non-linear function, denoted as the *squash* function, is applied on s_j to compute the final activation output v_j of capsule j. v_j is computed by

$$v_j = \frac{||s_j||^2}{1 + ||s_j||^2} \frac{s_j}{||s_j||} \tag{3}$$

The routing coefficients c_{ij} are normalized, which means for each i, $\sum_j c_{ij} = 1$. Furthermore, they are determined by *routing softmax*, whose initial logits b_{ij} are the log prior probabilities that capsule i should be routed to capsule j. The *routing softmax* is defined as

$$c_{ij} = \frac{exp(b_{ij})}{\sum_k exp(b_{ik})} \tag{4}$$

The b_{ij} can be learned in the iteration of dynamic routing. After each iteration, b_{ij} is updated by

$$b_{ij} = b_{ij} + \hat{u}_{j|i} \cdot v_j \tag{5}$$

We assume the capsules in the low layer, which represents low level features of a short subsequence, can only be routed to one capsule in the high layer, which represents high level features of a long subsequence. The reason is that if one capsule in the low layer can be routed to more than one capsules in the subsequent layer, the short subsequence represented by this capsule will not be a discriminative subsequence pattern. So, in the last iteration, we add an mask operation which only keep the maximum value of c_{ij} for each i, and set other c_{ij} to 0. The mask operation ensure that the capsule in the low layer only be routed to one capsule in the subsequent layer. The mask operation for each i is defined by

$$c_{ij} = \begin{cases} c_{ij} & j = arg\max(c_{ij}) \\ 0 & \text{others} \end{cases} \tag{6}$$

3.3 Classification

After obtaining the vector output of capsule layer, the norm of each vector, which represents the existing probability of the high level features, is computed. Thus, the new representation of the original time series T becomes:

$$\Phi(T) = (||v_1||, ||v_2||, \ldots, ||v_k||) \tag{7}$$

where v_i is the vector output of capsule layer.

Logistic Regression is applied on this new representation $\Phi(T)$ to estimate the probability of the input time series T is positive. For multi-class time series classification, we can replace logistic regression with softmax regression. During training, we use cross entropy as the loss function.

Table 1: Description of 44 UCR time series classification datasets

Dataset	Train	Test	Class	Length	Type
Adiac	390	391	37	176	Image
Beef	30	30	5	470	Spectro
CBF	30	900	3	128	Simulated
ChlorineCon	467	3840	3	166	Sensor
CinCECGTorso	40	1380	4	1639	Sensor
Coffee	28	28	2	286	Spectro
CricketX	390	390	12	300	Motion
CricketY	390	390	12	300	Motion
CricketZ	390	390	12	300	Motion
DiatomSizeR	16	306	4	345	Image
ECGFiveDays	23	861	2	136	ECG
FaceAll	560	1690	14	131	Image
FaceFour	24	88	4	350	Image
FacesUCR	200	2050	14	131	Image
fiftywords	450	455	50	270	Image
fish	175	175	7	463	Image
GunPoint	50	150	2	150	Motion
Haptics	155	308	5	1092	Motion
InlineSkate	100	550	7	1882	Motion
ItalyPower	67	1029	2	24	Sensor
Lightning2	60	61	2	637	Sensor
Lightning7	70	73	7	319	Sensor
MALLAT	55	2345	8	1024	Simulated
MedicalImages	381	760	10	99	Image
MoteStrain	20	1252	2	84	Sensor
NonInvThorax1	1800	1965	42	750	ECG
NonInvThorax2	1800	1965	42	750	ECG
OliveOil	30	30	4	570	Spectro
OSULeaf	200	242	6	427	Image
SonyAIBORobot	20	601	2	70	Sensor
SonyAIBORobotII	27	953	2	65	Sensor
StarLightCurves	1000	8236	3	1024	Sensor
SwedishLeaf	500	625	15	128	Image
Symbols	25	995	6	398	Image
SyntheticControl	300	300	6	60	Simulated
Trace	100	100	4	275	Sensor
TwoLeadECG	23	1139	2	82	ECG
TwoPatterns	1000	4000	4	128	Simulated
UWaveX	896	3582	8	315	Motion
UWaveY	896	3582	8	315	Motion
UWaveZ	896	3582	8	315	Motion
wafer	1000	6164	2	152	Sensor
WordSynonyms	267	638	25	270	Image
yoga	300	3000	2	426	Image

4 Experiments

To evaluate the performance of the proposed model, experiments are performed on UCR dataset [15] to compare our model with other leading methods. In addition, case study is applied to show the interpretability of our model.

4.1 UCR time series Classification

4.1.1 Dataset Introduction

The UCR time series classification dataset [15] contains 128 publicly available time series datasets which vary by the number of classes, dataset types, number of samples, and length of time series. Each data set was split into training and testing set by the provider. We choose 44 datasets which is a previous version of UCR dataset. Their characteristics are described in Table 1.

4.1.2 Evaluation Metric

We evaluate the models based on their accuracy on each dataset, in addition, we obtain an overall error rate using the Mean Per-Class Error (MPCE), as proposed in [11]. Let c_k denote the number of classes in the k^{th} dataset and $e_{i,k}$ denote the error rate of i^{th} model on the k^{th} dataset, K denote the number of datasets, then the MPCE score of the i^{th} model can be calculated as follows:

$$PCE_{i,k} = \frac{e_{i,k}}{c_k} \tag{8}$$

$$MPCE_i = \frac{1}{K} \sum_k PCE_k \tag{9}$$

4.1.3 Implementation Details

In these experiments, we followed the train/test dataset split provided by the UCR dataset. We implement our model using PyTorch [17]. We use 4 convolution layers without padding. The convolution kernel size is 3. We set the stride of the first convolution layer as 3, the last convolution layer as 2, others 1. The iteration number we use in dynamic routing is 3. The dimension of capsules and the number of capsules are set differently according to different datasets.

Since the large amount of parameters in our model, we apply *window slicing*, which is a data augmentation technique proposed in [10], on UCR datasets. For a time series $T = \{t_1, \ldots, t_n\}$, a slice is a subsequence of the original time series, defined as $S_{i:j} = \{t_i, t_{i+1}, \ldots, t_j\}, 1 \leq i \leq j \leq n$. For a given time series T of length n, the slicing window with length s will generate a set of $n - s + 1$ sliced time series:

$$Slicing(T, s) = \{S_{1:s}, S_{2:s+1}, \ldots, S_{n-s+1:n}\} \tag{10}$$

where all the time series in $Slicing(T, s)$ have the same label as the original time series T.

When training, slicing window is applied on all the time series in training dataset. When testing, slicing window is also applied on time series in testing dataset, our model will predict the label of each slice of the original time series, and then use a majority vote among all these slices to make the final prediction.

4.2 UCR time series Classification Results

In this section, we compare our proposed model with traditional methods and deep learning methods on UCR datasets separately.

4.2.1 Comparison with Traditional Methods

For traditional methods, we compare our method with ED [18], DTW [19], Fast shapelets (FS) [3], Bag-of-SFA-Symbols (BOSS) [7], SAX-VSM [5], learning shapelet (LS) [16], SE [8], COTE [9]. Among these methods, ED, DTW, FS and LS are shape based methods; SAX-VSM and BOSS are structure based methods; SE and COTE are ensemble based methods. And only FS, LS and SAX-VSM are interpretable methods.

The classification accuracy, average rank and MPCE score of our model and these 8 traditional methods are shown in Table 2. As shown in the table, our method achieves the best performance on 12 dataset of the 44 UCR datasets. Our model also achieves 3.5455 of the average rank which is only worse than COTE, which is an ensemble method without interpretability. Among the interpretable TSC methods, FS, LS and SAX-VSM, our model achieves the best performance regarding the average rank. Our model also achieve 0.0289 of the MPCE score which is only worse than LS.

Figure 2 shows the critical difference diagram, as proposed in [20]. The values shown on the figure are the average rank of each classifier. Bold lines indicate groups of classifiers which are not significantly different. As shown in Figure 2, our model is among the most accurate classifiers and is not significantly different from COTE.

4.2.2 Comparison with Deep Learning Methods

For deep learning methods, we compare our method with MCNN [10], MLP, FCN and ResNet [11]. The classification accuracies, average rank and MPCE score of our model and these 4 deep learning methods are shown in Table 3. As shown in the table, our method achieves the best performance on 11 dataset of the 44 UCR datasets. And from the critical difference diagram in Figure 3, we can know our model is not significantly different from MCNN, ResNet and FCN.

Table 2: Classification error of our model and traditional methods on 44 UCR datasets

Dataset	DTW	ED	FS	BOSS	SAX-VSM	LS	SE	COTE	Our model
Adiac	0.396	0.389	0.514	**0.220**	0.417	0.437	0.435	0.233	0.240
Beef	0.367	0.467	0.447	0.200	0.467	0.240	0.167	**0.133**	0.200
CBF	0.003	0.148	0.053	**0.000**	0.007	0.006	0.003	0.001	0.009
ChlorineCon	0.352	0.350	0.417	0.340	0.334	0.349	0.300	0.314	**0.157**
CinCECGTorso	0.349	0.103	0.174	0.125	0.344	0.167	0.154	0.064	**0.038**
Coffee	**0.000**	**0.000**	0.068	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
CricketX	0.246	0.423	0.527	0.259	0.308	0.209	0.218	**0.154**	0.236
CricketY	0.256	0.433	0.505	0.208	0.318	0.249	0.236	**0.167**	0.205
CricketZ	0.246	0.413	0.547	0.246	0.297	0.200	0.228	**0.128**	0.208
DiatomSizeR	**0.033**	0.065	0.117	0.046	0.121	**0.033**	0.124	0.082	0.042
ECGFiveDays	0.232	0.203	0.004	**0.000**	0.003	**0.000**	0.001	**0.000**	**0.000**
FaceAll	0.192	0.286	0.411	0.210	0.244	0.217	0.263	**0.105**	0.189
FaceFour	0.170	0.216	0.090	**0.000**	0.114	0.048	0.057	0.091	**0.000**
FacesUCR	0.095	0.231	0.328	**0.042**	0.100	0.059	0.087	0.057	0.093
fiftywords	0.310	0.369	0.489	0.301	0.374	0.232	0.281	**0.191**	0.220
fish	0.177	0.217	0.197	**0.011**	0.017	0.066	0.023	0.029	0.029
GunPoint	0.093	0.087	0.061	**0.000**	0.013	**0.000**	0.020	0.007	**0.000**
Haptics	0.623	0.630	0.616	0.536	0.575	0.532	0.523	**0.488**	0.578
InlineSkate	0.616	0.658	0.734	**0.511**	0.593	0.573	0.615	0.551	0.676
ItalyPower	0.050	0.045	0.095	0.053	0.089	**0.030**	0.048	0.036	0.499
Lightning2	**0.131**	0.246	0.295	0.148	0.230	0.177	0.344	0.164	0.230
Lightning7	0.274	0.425	0.403	0.342	0.342	**0.197**	0.260	0.247	0.219
MALLAT	0.066	0.086	**0.033**	0.058	0.199	0.046	0.060	0.036	0.052
MedicalImages	0.263	0.316	0.433	0.288	0.516	0.270	0.396	**0.258**	0.295
MoteStrain	0.165	0.121	0.217	**0.073**	0.117	0.087	0.109	0.085	0.133
NonInvThorax1	0.210	0.171	0.171	0.161		0.131	0.100	**0.093**	0.117
NonInvThorax2	0.135	0.120	0.120	0.101		0.089	0.097	**0.073**	0.095
OliveOil	0.167	0.133	0.213	**0.100**	0.133	0.560	**0.100**	**0.100**	0.223
OSULeaf	0.409	0.483	0.359	**0.012**	0.153	0.182	0.285	0.145	0.367
SonyAIBORobot	0.275	0.305	0.314	0.321	0.306	0.103	0.067	0.146	**0.043**
SonyAIBORobotII	0.169	0.141	0.215	0.098	0.126	0.082	0.115	**0.076**	0.083
StarLightCurves	0.093	0.151	0.060	**0.021**	0.108	0.033	0.024	0.031	0.039
SwedishLeaf	0.208	0.213	0.269	0.072	0.275	0.087	0.093	**0.046**	0.051
Symbols	0.050	0.100	0.068	**0.032**	0.089	0.036	0.114	0.046	0.041
SyntheticControl	0.007	0.120	0.081	0.030	0.013	0.007	0.017	**0.000**	0.003
Trace	**0.000**	0.240	0.002	0.002	**0.000**	**0.000**	0.020	0.010	**0.000**
TwoLeadECG	**0.000**	0.090	0.113	0.004	0.004	0.003	0.004	0.015	0.001
TwoPatterns	0.096	0.253	0.090	0.016	0.011	0.003	0.059	**0.000**	**0.000**
UWaveX	0.272	0.261	0.293	0.241	0.324	0.200	0.216	0.196	**0.159**
UWaveY	0.366	0.338	0.392	0.313	0.364	0.287	0.303	0.267	**0.233**
UWaveZ	0.342	0.350	0.364	0.312	0.357	0.268	0.273	0.265	**0.229**
wafer	0.020	0.005	0.004	**0.001**	0.002	0.004	0.002	**0.001**	0.006
WordSynonyms	0.351	0.382	0.563	0.345	0.436	0.340	0.403	**0.266**	0.323
yoga	0.164	0.170	0.249	**0.081**	0.151	0.150	0.195	0.113	0.109
WIN	5	1	1	16	2	7	2	**19**	12
AVG Rank	5.9318	7.1477	7.6591	3.6136	6.3068	3.5909	4.7614	**2.4432**	3.5455
MPCE	0.0397	0.0467	0.4740	0.0256	0.0364	0.0274	0.0295	**0.0226**	0.0289

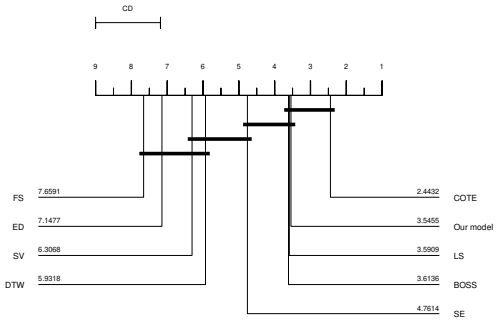

Figure 2: Critical difference diagram over the average rank of our model and 8 traditional methods

Table 3: Classification error of our model and deep learning methods on 44 UCR datasets

Dataset	MCNN	MLP	FCN	ResNet	Our model
Adiac	0.231	0.248	**0.143**	0.174	0.240
Beef	0.367	**0.167**	0.250	0.233	0.200
CBF	0.002	0.140	**0.000**	0.006	0.009
ChlorineCon	0.203	**0.128**	0.157	0.172	0.157
CinCECGTorso	0.058	0.158	0.187	0.229	**0.038**
Coffee	0.036	**0.000**	**0.000**	**0.000**	**0.000**
CricketX	0.182	0.431	0.185	**0.179**	0.236
CricketY	**0.154**	0.405	0.208	0.195	0.205
CricketZ	**0.142**	0.408	0.187	0.187	0.208
DiatomSizeR	**0.023**	0.036	0.070	0.069	0.042
ECGFiveDays	**0.000**	0.030	0.015	0.045	**0.000**
FaceAll	0.235	0.115	**0.071**	0.166	0.189
FaceFour	**0.000**	0.170	0.068	0.068	**0.000**
FacesUCR	0.063	0.185	0.052	**0.042**	0.093
fiftywords	**0.190**	0.288	0.321	0.273	0.220
fish	0.051	0.126	0.029	**0.011**	0.029
GunPoint	**0.000**	0.067	**0.000**	0.007	**0.000**
Haptics	0.530	0.539	**0.449**	0.495	0.578
InlineSkate	0.618	0.649	**0.589**	0.635	0.676
ItalyPower	**0.030**	0.034	**0.030**	0.040	0.499
Lightning2	**0.164**	0.279	0.197	0.246	0.230
Lightning7	0.219	0.356	**0.137**	0.164	0.219
MALLAT	0.057	0.064	**0.020**	0.021	0.052
MedicalImages	0.260	0.271	**0.208**	0.228	0.295
MoteStrain	0.079	0.131	**0.050**	0.105	0.133
NonInvThorax1	0.064	0.058	**0.039**	0.052	0.117
NonInvThorax2	0.060	0.057	**0.045**	0.049	0.095
OliveOil	**0.133**	0.600	0.167	**0.133**	0.223
OSULeaf	0.271	0.430	**0.012**	0.021	0.367
SonyAIBORobot	0.230	0.273	0.032	**0.015**	0.043
SonyAIBORobotII	0.070	0.161	**0.038**	**0.038**	0.083
StarLightCurves	**0.023**	0.043	0.033	0.029	0.039
SwedishLeaf	0.066	0.107	**0.034**	0.042	0.051
Symbols	0.049	0.147	**0.038**	0.128	0.041
SyntheticControl	0.003	0.050	0.010	**0.000**	0.003
Trace	**0.000**	0.180	**0.000**	**0.000**	**0.000**
TwoLeadECG	0.001	0.147	**0.000**	**0.000**	0.001
TwoPatterns	0.002	0.114	0.103	**0.000**	**0.000**
UWaveX	0.180	0.232	0.246	0.213	**0.159**
UWaveY	0.268	0.297	0.275	0.332	**0.233**
UWaveZ	0.232	0.295	0.271	0.245	**0.229**
wafer	**0.002**	0.004	0.003	0.003	0.006
WordSynonyms	**0.276**	0.406	0.420	0.368	0.323
yoga	0.112	0.145	0.155	0.142	**0.109**
WIN	14	3	**20**	11	11
AVG Rank	2.5909	4.1932	**2.4773**	2.6250	3.1136
MPCE	0.0241	0.0401	**0.0214**	0.0231	0.0289

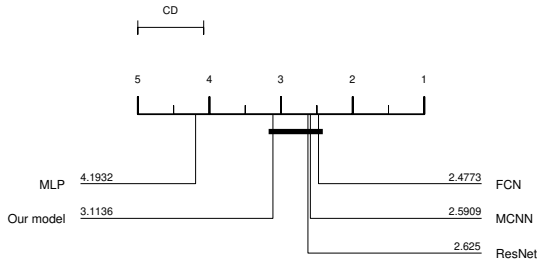

Figure 3: Critical difference diagram over the average rank of our model and 4 deep learning methods

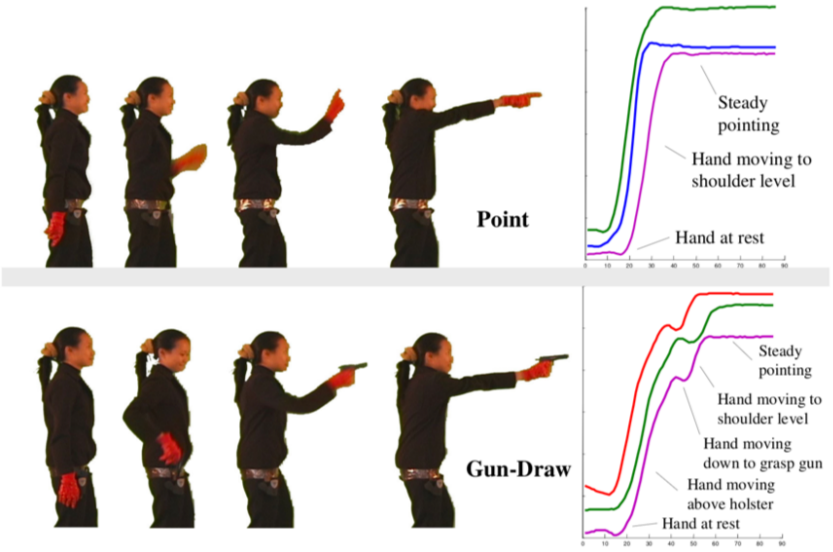

Figure 4: GunPoint Dataset

4.3 Case Study

We use case study to exhibit the interpretability of our model. This case study is based on GunPoint dataset shown in Figure 4. For GunPoint dataset, this dataset takes into account of the physical movements of each participant. The dataset consists of two classes: Gun-Draw and Point. For Gun-Draw class, people have their hands by their sides. They draw a replicate gun from a hip-mounted holster, point it at a target for approximately one second, then return the gun to the holster, and their hands to their sides. For Point class, people have their hands by their sides. They point with their index fingers to a target for approximately one second, and then return their hands to their sides. For both classes, the track of the centroid of people's right hand constitute the time series. So, the main difference between these two classes is in the period of drawing gun and returning gun. We exhibit the interpretability of our model in two aspects, shapelet and point.

4.3.1 Shapelet

In order to exhibit the interpretability of DRLR, for each capsule which is the input of logistic regression, we backtrack the routing process and convolution to find the corresponding points in the raw time series which are routed to this capsule. We only show the backtracked points of active capsules which are the capsules with norm above 0.1. The result is shown in Figure 5 and Figure 6. Among figure (a) to (f) in Figure 5 and Figure 6, the points with purple color are general points in the time series, and the points with red color are the points which are routed to the corresponding capsule. The norm of output vector of the capsule and the corresponding weight of Logistic Regression for this capsule is also shown in these figures.

Every capsule can represent a specific shapelet (dataset shapelet) which are important for the whole dataset, the points which are grouped into this capsule is the shapelet detected in this time series. The norm represents the similarity between the dataset shapelet and the detected shapelet in this time series. The weight can represent the importance of the dataset shapelet. The value, $norm * weight$ of this capsule, can represent the contribution of this detected shapelet for the final decision. In Figure 6 (e), this capsule find the shapelet which are exactly the range of drawing gun and returning gun. These results are very similar to the best pattern found for GunPoint dataset in SAX-VSM [5]. Unlike other shapelet based method which usually find the dataset shapelet, the capsule in our model can find different shapelets for different time series even they are from the same class. It is more reasonable to find different shapelets for different time series. We suspect this is one of the reasons our method can outperform their methods.

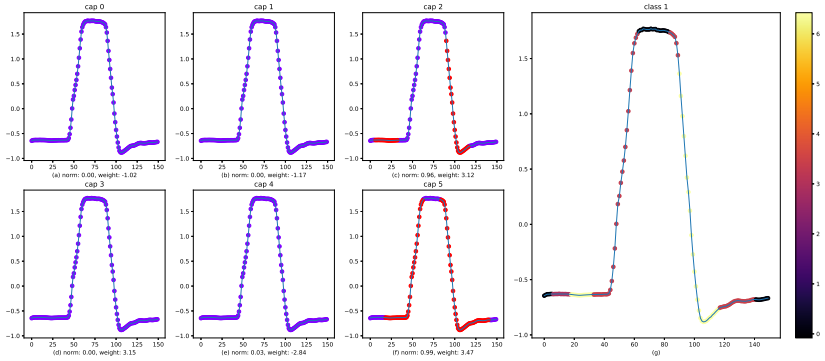

Figure 5: Point time series

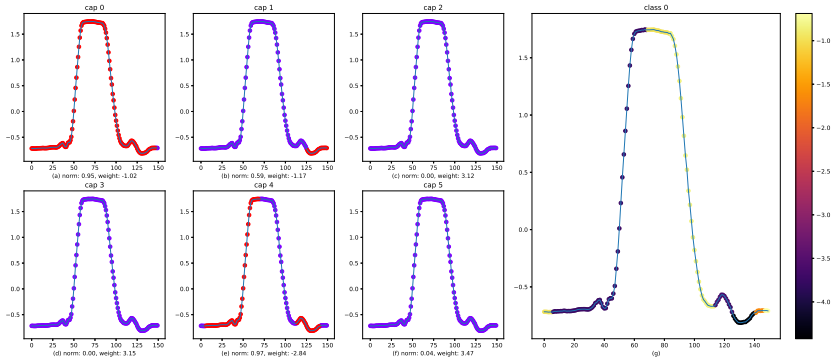

Figure 6: Gun-Draw time series

4.3.2 Point

Since there are usually overlap between the shapelets found by the capsules, we combine all the found shapelets to find the importance of each points in the time series. We initialize value $w_i = 0$ for each point i, then for each capsule j and each point i, if point i is routed to capsule j, we add the value, $norm_j * weight_j$, to w_i. Then the value w_i can represent the importance of point i for making the final decision. w_i is computed as

$$w_i = \sum_{point(i) \in cap(j)} norm_j * weight_j \tag{11}$$

The figure (g) in the Figure 6 and Figure 5 show the point importance w_i of each time series. In Figure 5 (g), it is a Point time series, we can find the most important points of the time series is in the region of $(90, 115)$. This region represents people return their hands to their sides without returning gun. While in Figure 6 (g), it is a Gun-Draw time series, we can find the most important points for this time series is in the interval of $(20, 60)$ and $(110, 140)$. These two regions represent people drawing gun from the holster and returning gun to the holster.

5 Conclusion

In this project, we propose DRLR model based on convolution, dynamic routing and logistic regression for interpretable time series classification. The experiments on UCR dataset show the competitive

performance of our DRLR model compared with other state-of-the-art methods. More importantly, the case study also shows the interpretability of our model. To the best of our knowledge, our model is the first interpretable time series model based on dynamic routing, we hope DRLR will inspire more research on interpretable time series classification.

References

[1] Hiroaki Sakoe, Seibi Chiba, A Waibel, and KF Lee. Dynamic programming algorithm optimization for spoken word recognition. *Readings in speech recognition*, 159:224, 1990.

[2] Lexiang Ye and Eamonn Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 947–956. ACM, 2009.

[3] Thanawin Rakthanmanon and Eamonn Keogh. Fast shapelets: A scalable algorithm for discovering time series shapelets. In *proceedings of the 2013 SIAM International Conference on Data Mining*, pages 668–676. SIAM, 2013.

[4] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2):107–144, 2007.

[5] Pavel Senin and Sergey Malinchik. Sax-vsm: Interpretable time series classification using sax and vector space model. In *2013 IEEE 13th international conference on data mining*, pages 1175–1180. IEEE, 2013.

[6] Patrick Schäfer and Mikael Högqvist. Sfa: a symbolic fourier approximation and index for similarity search in high dimensional datasets. In *Proceedings of the 15th International Conference on Extending Database Technology*, pages 516–527. ACM, 2012.

[7] Patrick Schäfer. The boss is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29(6):1505–1530, 2015.

[8] Jason Lines, Jon Hills, and Anthony Bagnall. The collective of transformation-based ensembles for time-series classification. *Ueaeprints. Uea. Ac. Uk*, 2014.

[9] Anthony Bagnall, Jason Lines, Jon Hills, and Aaron Bostrom. Time-series classification with cote: the collective of transformation-based ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2522–2535, 2015.

[10] Zhicheng Cui, Wenlin Chen, and Yixin Chen. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995*, 2016.

[11] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866, 2017.

[14] Wei Zhao, Jianbo Ye, Min Yang, Zeyang Lei, Suofei Zhang, and Zhou Zhao. Investigating capsule networks with dynamic routing for text classification. *arXiv preprint arXiv:1804.00538*, 2018.

[15] Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive, October 2018. `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`.

[16] Josif Grabocka, Nicolas Schilling, Martin Wistuba, and Lars Schmidt-Thieme. Learning time-series shapelets. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 392–401. ACM, 2014.

[17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[18] Christos Faloutsos, Mudumbai Ranganathan, and Yannis Manolopoulos. *Fast subsequence matching in time-series databases*, volume 23. ACM, 1994.

[19] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[20] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.