DATA MINING 2
Time Series Classification

Riccardo Guidotti

a.a. 2022/2023
Time Series Classification

• Given a set $X$ of $n$ time series, $X = \{x_1, x_2, \ldots, x_n\}$, each time series has $m$ ordered values $x_i = \langle x_{i1}, x_{i2}, \ldots, x_{im} \rangle$ and a class value $c_i$.

• The objective is to find a function $f$ that maps from the space of possible time series to the space of possible class values.

• Generally, it is assumed that all the TS have the same length $m$. 
KNN Classification

- The most widely used and effective approach for TSC consists in using KNN on the raw time series.

- Pros:
  - Simple
  - Dynamic Time Warping gives much better results than Euclidean distance on many problems.

- Cons:
  - KNN is a lazy classifier and computationally expensive on its own
  - Dynamic Time Warping is very very slow to calculate
Shapelet-based Classification

1. Represent a TS as a vector of distances with representative subsequences, namely shapelets.

2. Use it to as input for machine learning classifiers.
Shapelet-based Classifiers

- Time Series Dataset
- Shapelet Finder
- Shapelet Transformer
- Shapelet Dataset
- ML Model

Symbols:
- $N \times M$
- $K$
- $N \times K$
Shapelets are TS subsequences which are maximally representative of a class.

Shapelets can provide interpretable results, which may help domain practitioners better understand their data.

Shapelets can be significantly more accurate/robust because they are local features, whereas most other state-of-the-art TS classifiers consider global features.
Extract Subsequences of all Possible Lengths
Extract Subsequences of all Possible Lengths
Extract Subsequences of all Possible Lengths
Extract Subsequences of all Possible Lengths
Extract Subsequences of all Possible Lengths
Distance with a Subsequence

- Distance from the TS to the subsequence $SubsequenceDist(T, S)$ is a distance function that takes time series $T$ and subsequence $S$ as inputs and returns a nonnegative value $d$, which is the distance from $T$ to $S$.

- $SubsequenceDist(T, S) = \min(Dist(S, S'))$, for $S' \in S_T^{|S|}$

- where $S_T^{|S|}$ is the set of all possible subsequences of $T$

- Intuitively, it is the distance between $S$ and its best matching location in $T$. 
Testing The Utility of a Candidate Shapelet

• Arrange the TSs in the dataset $D$ based on the distance from the candidate.

• Find the optimal split point that maximizes the information gain (same as for Decision Tree classifiers)

• Pick the candidate achieving best utility as the shapelet
• A TS dataset $D$ consists of two classes, A and B.
• Given that the proportion of objects in class A is $p(A)$ and the proportion of objects in class B is $p(B)$,
• The **Entropy** of $D$ is: $I(D) = -p(A)\log(p(A)) - p(B)\log(p(B))$.
• Given a strategy that divides the $D$ into two subsets $D_1$ and $D_2$, the information remaining in the dataset after splitting is defined by the weighted average entropy of each subset.
• If the fraction of objects in $D_1$ is $f(D_1)$ and in $D_2$ is $f(D_2)$,
• The total entropy of $D$ after splitting is $\hat{I}(D) = f(D_1)I(D_1) + f(D_2)I(D_2)$. 
Information Gain

- Given a certain split strategy \( sp \) which divides \( D \) into two subsets \( D_1 \) and \( D_2 \), the entropy before and after splitting is \( I(D) \) and \( \hat{I}(D) \).
- The **information gain** for this splitting rule is:
  \[
  \text{Gain}(sp) = I(D) - \hat{I}(D) = I(D) - f(D_1)I(D_1) + f(D_2)I(D_2).
  \]
- We use the distance from \( T \) to a shapelet \( S \) as the splitting rule \( sp \).
Problem

- The total number of candidate is

\[
\sum_{l=\text{MINLEN}}^{\text{MAXLEN}} \sum_{T_i \in D} (|T_i| - l + 1)
\]

- For each candidate you have to compute the distance between this candidate and each training sample

- For instance
  - 200 instances with length 275
  - 7,480,200 shapelet candidates
Speedup

• Distance calculations form TSs to shapelet candidates is expensive.
• Reduce the time in two ways
  • Distance Early Abandon
    • reduce the distance computation time between two TS
  • Admissible Entropy Pruning
    • reduce the number of distance calculations
Distance Early Abandon

• We only need the minimum distance.

• Method
  • Keep the best-so-far distance
  • Abandon the calculation if the current distance is larger than best-so-far.
Admissible Entropy Pruning

• We only need the best shapelet for each class

• For a candidate shapelet
  • We do not need to calculate the distance for each training sample
  • After calculating some training samples, the upper bound of information gain < best candidate shapelet
  • Stop calculation
  • Try next candidate
Shapelet Summary

1. Extract all possible subsequences of a set given lengths (candidate shapelets)

2. For each candidate shapelet
   1. Calculate the distance with each time series keeping the minimum distance (best alignment)
   2. Evaluate the discriminatory effect of the shapelet through the Information Gain

3. Return the $k$ best shapelets with the highest Information Gain.

4. Transform a dataset and train a ML model.
An Alternative Way for Extracting Shapelets

• The minimum distances (M) between Ts and Shapelets can be used as predictors to approximate the TSs label (Y) using a linear model (W):

\[ \hat{Y}_i = W_0 + \sum_{k=1}^{K} M_{i,k} W_k, \quad \forall i \in \{1, \ldots, I\} \]

• A logistic regression loss can measure the quality of the prediction:

\[ \mathcal{L}(Y, \hat{Y}) = -Y \ln \sigma(\hat{Y}) - (1 - Y) \ln \left( 1 - \sigma(\hat{Y}) \right) \]

• The objective is to minimize a regularized loss function across all the instances (I) :

\[ \arg\min_{S, W} \mathcal{F}(S, W) = \arg\min_{S, W} \sum_{i=1}^{I} \mathcal{L}(Y_i, \hat{Y}_i) + \lambda_W \|W\|^2 \]

• We can find the optimal shapelet for the objective function in a NN fashion by updating the shapelets in the minimum direction of the objective, hence the first gradient. Similarly, the weights can be jointly updated towards minimizing the objective function.
Motif/Shapelet Summary

• A **motif** is a repeated pattern/subsequence in a given TS.

• A **shapelet** is a pattern/subsequence which is maximally representative of a class with respect to a given dataset of TSs.
References

• Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View that Includes Motifs, Discords and Shapelets. Chin-Chia Michael Yeh et al. 1997


References


• Compression-based data mining of sequential data. Eamonn Keogh et al. 2007.
TSC State-of-The-Art

A special thank to Francesco Spinnato for the slides
ResNet

• Three consecutive blocks, comprised of three convolutional layers, connected by residual ‘shortcut’ connections.
• The blocks are followed by global average pooling and softmax layers to form features and subsequent predictions.
Convolution Layer

- 32x32x3 image
- 5x5x3 filter

Convolve (slide) over all spatial locations

Activation map

Dimensions:
- 32x32x3 image
- 5x5x3 filter
- 28x28 activation map
Pooling Layer

• Makes the representations smaller and more manageable
• Operates over each activation map independently
MaxPooling and AvgPooling
InceptionTime

Neural network ensemble consisting of five Inception networks. For each inception network:

• three Inception modules (6 blocks by default)
• global averaging pooling
• fully-connected layer with the softmax activation function.

Each Inception module consists of convolutions with kernels of several sizes followed by batch normalization and the rectified linear unit activation function.
InceptionTime

- Input time series
- Convolution
- Residual connections
- Global average pooling
- Fully connected
- Output classes

Channels vs. Time
TapNet

Draws on the strengths of both traditional and deep learning approaches:

- **deep learning approaches** -> excel at learning low dimensional features without the need for embedded domain knowledge, whereas
- **traditional approaches** -> work well on small datasets.

3 distinct modules:

- Random Dimension Permutation: produce groups of randomly selected dimensions with the intention of increasing the likelihood of learning how combinations of dimension values effect class value.
- Multivariate Time Series Encoding:
  - 3 sets of 1d convolutional layers followed by batch normalisation
  - the raw data is also passed through an LSTM and global pooling layer
- Attentional Prototype Learning: used for unlabelled data
Canonical Interval Forest (CIF)

Ensemble of time series tree classifiers built using the 22 Canonical Time-Series Characteristics (Catch22) features and simple summary statistics (mean, stdev, slope).

For each tree, CIF:

• samples $k$ time series intervals of random position and length;
• subsamples 8 of the 25 features randomly;
• calculates the features for each interval, concatenates them to form a new data set;
• builds a decision tree on the feature-transformed dataset.
ROCKET

ROCKET (Random Convolutional Kernel Transform) uses a large number of random convolutional kernels to transform the time series:

• all the parameters of all the kernels are randomly generated from fixed distributions;

• the transformed features are used to train a linear classifier (Logistic Regression or Ridge Regression Classifier);

• the combination of Rocket and logistic regression forms a single-layer convolution with random kernel weights with a trained softmax layer.
CNNs use trainable filters/kernels optimized by stochastic gradient descent to find patterns in the input data. Rocket differs in the following ways:

- Only a single layer containing a very large number of random kernels.
- Variety of kernels: each kernel has random length, dilation, and padding, weights and biases.
Dilated Convolution Kernels

3x3 Kernel

Dilated Convolution (Rate = 1)

Dilated Convolution (Rate = 2)

Dilated Convolution (Rate = 3)
ROCKET vs. CNN

• In CNNs kernel dilation increases exponentially with depth. Rocket sample dilation randomly for each kernel, capturing patterns at different frequencies and scales.

• Rocket uses the maximum value of the resulting feature maps (~global max pooling), and the proportion of positive values (proportion of the input which matches a given pattern).

• The only hyperparameter for Rocket is the number of kernels, $k$.
  • $k$ handles the trade-off between classification accuracy and computation time
MiniRocket removes almost all randomness from Rocket, and dramatically speeds up the transform.

- **Length**: uses kernels of length 9.
- **Weights**: restricted to two values, $\alpha = -1$ and $\beta = 2$.
- **Kernels**: there are 512 possible two-valued kernels of length 9. Only subset of 84 is used.
- **Bias**: drawn from the quantiles of the convolution output for the entire training set (rather than a single, randomly-selected training example)
- **Dilation**: Each kernel is assigned the same fixed set of dilations, adjusted to the length of the input time series. The maximum number of dilations per kernel is 32
- **Padding**: half the kernel/dilation combinations use padding, and half do not.
- **Features**: only proportion of positive values.
COTE / HIVE-COTE / TS-CHIEF

• Collective of Transformation-Based Ensembles (COTE) combines 35 classifiers over four data representations (similarity measures, shapelet-transform, autocorrelation features, power spectrum).

• Hierarchical Vote Collective of Transformation-Based Ensembles (HIVE-COTE) is an extension of COTE including more classifiers and a hierarchical voting procedure.

• Time Series Combination of Heterogeneous and Integrated Embedding Forest (TS-CHIEF) builds a random forest of decision trees whose splitting functions are time series specific and based on similarity measures, dictionary (bag-of-words) representations, and interval-based transformations.
MR-SEQL

- The data is discretized into sequences of words via either Symbolic Aggregate Approximation (SAX) or SFA, using a sliding window.
- The most discriminative symbols are extracted using a SEQuence Learner algorithm.
- The dataset is transformed in presence/absence of subsequences (similar to a shapelet transform)
- A linear (interpretable) model is trained on this new representation
MR-SEQL
Fig. 10  Average difference in accuracy to DTW$_D$ versus train time for 9 MTSC algorithms
Ranking Multivariate TSC algorithms

(a) Accuracy

(b) AUROC

(c) Balanced Accuracy

(d) F1
References


