Assignment 3 (ML for TS) - MVA 2023/2024

Halvard Bariller halvard.bariller@ens-paris-saclay.fr
Farah Ellouze farah.ellouze@ens-paris-saclay.fr

December 29, 2023

1 Introduction

Objective. The goal is to implement (i) a signal processing pipeline with a change-point detection method and (ii) wavelets for graph signals.

Warning and advice.

- Use code from the tutorials as well as from other sources. Do not code yourself well-known procedures (e.g. cross validation or k-means), use an existing implementation.
- The associated notebook contains some hints and several helper functions.
- Be concise. Answers are not expected to be longer than a few sentences (omitting calculations).

Instructions.

- Fill in your names and emails at the top of the document.
- Hand in your report (one per pair of students) by Sunday 31st December 11:59 PM.
- Rename your report and notebook as follows:
  `FirstnameLastname1_FirstnameLastname1.pdf` and
  `FirstnameLastname2_FirstnameLastname2.ipynb`.
  For instance, `LaurentOudre_CharlesTruong.pdf`.
- Upload your report (PDF file) and notebook (IPYNB file) using this link:
  docs.google.com/forms/d/e/1FAIpQLScqLsYuKeQbsDEOie5OqpOH7YwCnWmudzApMC005HvxOaOv

2 Dual-tone multi-frequency signaling (DTMF)

Dual-tone multi-frequency signaling is a procedure to encode symbols using an audio signal. The possible symbols are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, *, #, A, B, C, and D. A symbol is represented by a sum of cosine waves: for $t = 0, 1, \ldots, T-1$,

$$y_t = \cos(2\pi f_1 t / f_s) + \cos(2\pi f_2 t / f_s)$$

where each combination of (f_1, f_2) represents a symbols. The first frequency has four different levels (low frequencies), and the second frequency has four other levels (high frequencies); there are 16 possible combinations. In the notebook, you can find an example symbol sequence encoded with sound and corrupted by noise (white noise and a distorted sound).

Question 1

Design a procedure that takes a sound signal as input and outputs the sequence of symbols. To that end, you can use the provided training set. The signals have a varying number of symbols with a varying duration. There is a brief silence between each symbol.

Describe in 5 to 10 lines your methodology and the calibration procedure (give the hyperparameter values). Hint: use the time-frequency representation of the signals, apply a change-point detection algorithm to find the starts and ends of the symbols and silences, and then classify each segment.

Answer 1

We clearly observe on the below example (Figure 1) that the time-frequency representation is particularly relevant to the task we are trying to perform. Indeed, we can see distinctly identify each of the components of the signal:

- there are 10 different symbols, each of them being composed of a lower frequency and a higher frequency, and they are all separated by a silence
- we distinguish the distorted sound corrupting the signal, represented by a sinusoid centered around 2kHz
- we can observe the white noise spread out over the whole frequency range

Reflecting on these visualisations, we can establish the method we will use to perform symbol identification. The first step will be to determine the time-windows in a signal that contain a symbol. Once the symbols are located in time, we can train a classifier on relevant features to associate the proper symbol to the considered time-window .

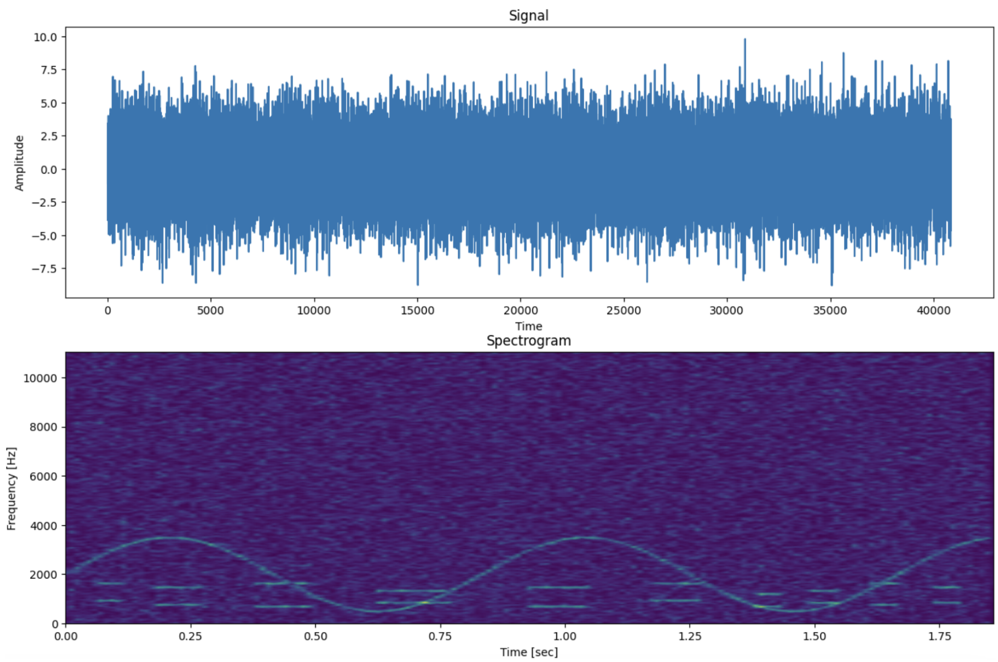


Figure 1: Signal and time-frequency representation

The signals being corrupted, one single model seemed to be not enough to detect each of the symbols. To circumvent this difficulty, we performed several change-point detection methods and trained a different model for each of them. By doing so with a wide range of hyperparameters, we built a variety of experts with different strengths and weaknesses, and aggregated their results so as to leverage their diversity.

We considered three types of signal for change-point detection:

- **Energy**: We designed an Energy function specific to our problem, with the idea that during a silence the energy should suddenly decrease. At each time-step, we compute the frequency-energy by summing the products of the represented frequencies by the corresponding intensities (given by the spectrogram). At each time-step, we compute the activation-energy as the number of values in the spectrogram above a certain threshold. The energy is computed as the sum of the normalized frequency-energy and the normalized activation-energy (see Figure 2)
- **Filtered Energy**: Energy obtained after applying a Butterworth bandpass filter to the original signal (to decrease the noise)
- **Filtered Spectrogram**: Spectrogram obtained after applying a Butterworth bandpass filter to the original signal (to decrease the noise)

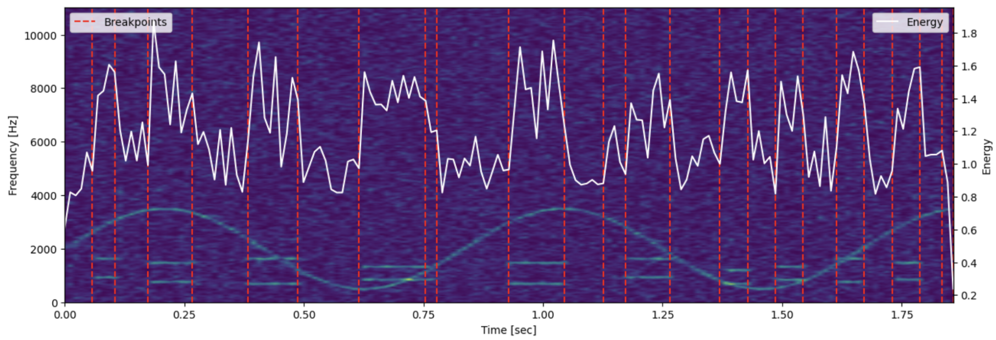


Figure 2: Signal Spectrogram and Energy

We used the Linearly Penalized Segmentation method `Pelt` implemented in the package *ruptures* [Truong et al., 2020] with various existing losses. We implemented an additional cost function `SilenceCost` using the negative energy as loss.
Before training the classifiers, we applied a few preprocessing techniques:

1. Given a list of detected breakpoints, we identified the silences as the period where the energy was below the mean energy of the signal
2. After detecting the silences, we averaged the contributions per frequency-levels (obtained with the spectrogram) over each time-window containing a symbol, and extracted the $k = 2$ frequency-levels contributing the most over the time-window (averaging helped to diminish the impact of the noise)
3. We merged the windows containing symbols if the windows were contiguous (no silence detected in-between) and if the two frequencies contributing the most to each window were the same
4. To use these extracted frequencies as features for our classifiers, we sorted them and performed One-Hot encoding

The models hyperparameters are gathered in Table 1.

Table 1: Models and hyperparameters

	Change-Point Detection					Classifier	
	Signal	Model	Min_size	Jump	Penalty	Loss	Penalty
Learner 1	Energy	L2	5	2	0.15	Log-loss	Ridge
Learner 2	Energy	L1	5	1	0.25	Log-loss	ElasticNet
Learner 3	Filtered Energy	L2	5	1	0.15	Log-loss	ElasticNet
Learner 4	Filtered Energy	L1	5	1	0.15	Log-loss	LASSO
Learner 5	Filtered Spectrogram	SilenceCost	5	1	0.8	Log-loss	ElasticNet
Learner 6	Filtered Spectrogram	L2	5	1	0.6	Log-loss	ElasticNet

The prediction pipeline is summarized in Figure 3.

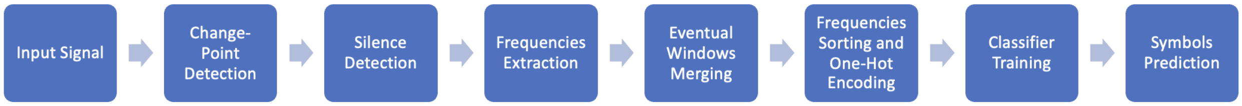


Figure 3: Prediction Pipeline

With this method, for each signal we get 6 different outputs composed of the predictions of each classifier for each change-point detected with the corresponding method. As learners might detect different change-points, we might have outputs of varying lengths. Our aggregation addresses this issue by first realigning the sequences of symbols on the longest one that the models output using an alignment algorithm derived from Needleman and Wunsch [1970]. We then proceed to a majority vote symbol by symbol in order to get the prediction of the whole sequence.

We used cross-validation in order to tune the hyperparameters.

Miscellaneous hyperparameters:

- Activation-energy threshold: $t = 0.4$
- Butterworth filter: $low = 350$, $high = 3500$, $order = 10$
- Number of relevant frequencies to extract: $k = 2$

Question 2

What are the two symbolic sequences encoded in the test set?

Answer 2

- Sequence 1: 721C99
- Sequence 2: 1#2#

3 Wavelet transform for graph signals

Let G be a graph defined a set of n nodes V and a set of edges E. A specific node is denoted by v and a specific edge, by e. The eigenvalues and eigenvectors of the graph Laplacian L are $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ and $u_1, u_2, \ldots, u_n$ respectively.

For a signal $f \in \mathbb{R}^n$, the Graph Wavelet Transform (GWT) of f is $W_f : \{1, \ldots, M\} \times V \longrightarrow \mathbb{R}$:

$$W_f(m, v) := \sum_{l=1}^{n} \hat{g}_m(\lambda_l) \hat{f}_l u_l(v) \tag{1}$$

where $\hat{f} = [\hat{f}_1, \ldots, \hat{f}_n]$ is the Fourier transform of f and $\hat{g}_m$ are M kernel functions. The number M of scales is a user-defined parameter and is set to $M := 9$ in the following. Several designs are available for the $\hat{g}_m$; here, we use the Spectrum Adapted Graph Wavelets (SAGW). Formally, each kernel $\hat{g}_m$ is such that

$$\hat{g}_m(\lambda) := \hat{g}^U(\lambda - am) \quad (0 \leq \lambda \leq \lambda_n) \tag{2}$$

where $a := \lambda_n / (M + 1 - R)$,

$$\hat{g}^U(\lambda) := \frac{1}{2}\left[1 + \cos\left(2\pi\left(\frac{\lambda}{aR} + \frac{1}{2}\right)\right)\right] \mathbb{1}(-Ra \leq \lambda < 0) \tag{3}$$

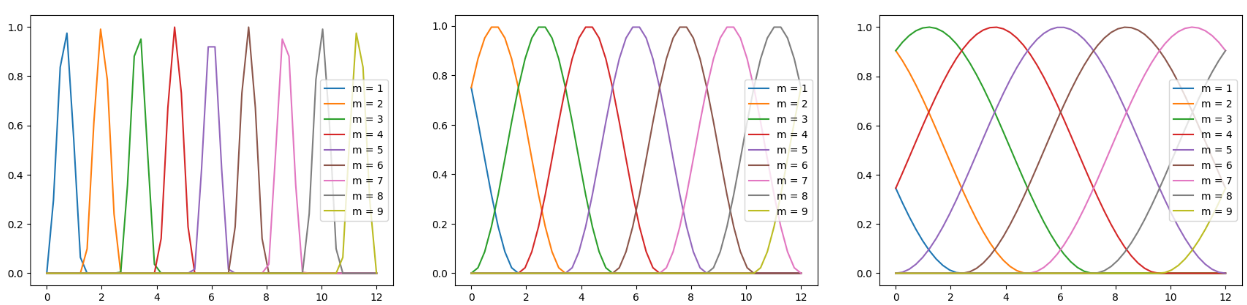
and $R > 0$ is defined by the user.

Question 3

Plot the kernel functions $\hat{g}_m$ for $R = 1$, $R = 3$ and $R = 5$ (take $\lambda_n = 12$) on Figure 4. What is the influence of R?

Answer 3

- Different values of R will result in different wavelet shapes. We notice in the Figure 4 below that for $R = 1$ the kernels are not overlapping. As R increases, the overlap also increases : this will help focus on specific areas of the data and smooth out the signal.



(a) $R = 1$ (b) $R = 3$ (c) $R = 5$

Figure 4: The SAGW kernels functions

We will study the Molene data set (the one we used in the last tutorial). The signal is the temperature.

Question 4

Construct the graph using the distance matrix and exponential smoothing (use the median heuristics for the bandwidth parameter).

- Remove all stations with missing values in the temperature.
- Choose the minimum threshold so that the network is connected and the average degree is at least 3.
- What is the time where the signal is the least smooth?
- What is the time where the signal is the smoothest?

Answer 4

The stations with missing values are Arzal, Batz, Beg-Meil, Brest-Guipavas, Brignogan, Camaret, Landivisiau, Lannaero, Lanveoc, Ouessant-Stiff, Plouay-SA, Ploudalmezeau, Plougonvelin, Quimper, Riec sur Belon, Sizun, St Nazaire-Montoir, Vannes-Meucon.

The threshold is equal to 0.83166.

The signal is the least smooth at 2014-01-21 06:00:00.

The signal is the smoothest at 2014-01-24 19:00:00.

Question 5

(For the remainder, set $R = 3$ for all wavelet transforms.)

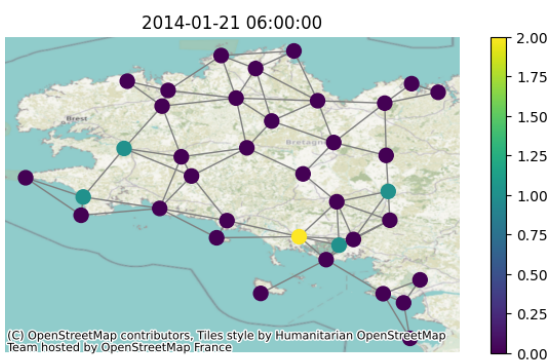
For each node v, the vector $[W_f(1, v), W_f(2, v), \ldots, W_f(M, v)]$ can be used as a vector of features. We can for instance classify nodes into low/medium/high frequency:

- a node is considered low frequency if the scales $m \in \{1, 2, 3\}$ contain most of the energy,
- a node is considered medium frequency if the scales $m \in \{4, 5, 6\}$ contain most of the energy,
- a node is considered high frequency if the scales $m \in \{6, 7, 9\}$ contain most of the energy.

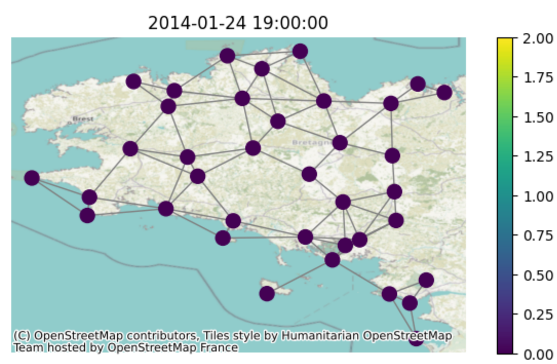
For both signals from the previous question (smoothest and least smooth) as well as the first available timestamp, apply this procedure and display on the map the result (one colour per class).

Answer 5

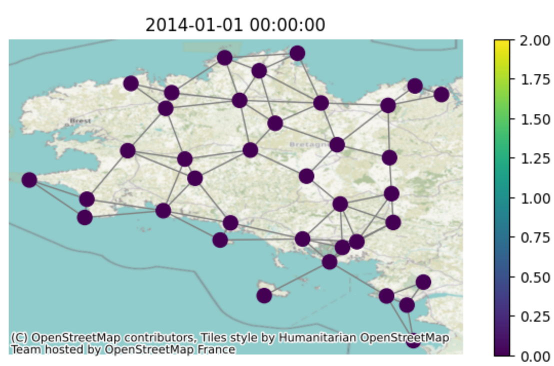
- Label 0 : Low Frequency
- Label 1 : Medium Frequency
- Label 2 : High Frequency



(a) Least smooth signal



(b) Smoothest signal



(c) First available timestamp

Figure 5: Classification of nodes into low/medium/high frequency

For the least smooth signal, the majority of nodes belong to the low frequency class with some

from the medium frequency class and lesser from the high frequency class.
For the smoothest signal as well as for the first available timestamp's signal, all the nodes are classified as low frequency.

Question 6

Display the average temperature and for each timestamp, adapt the marker colour to the majority class present in the graph (see notebook for more details).

Answer 6

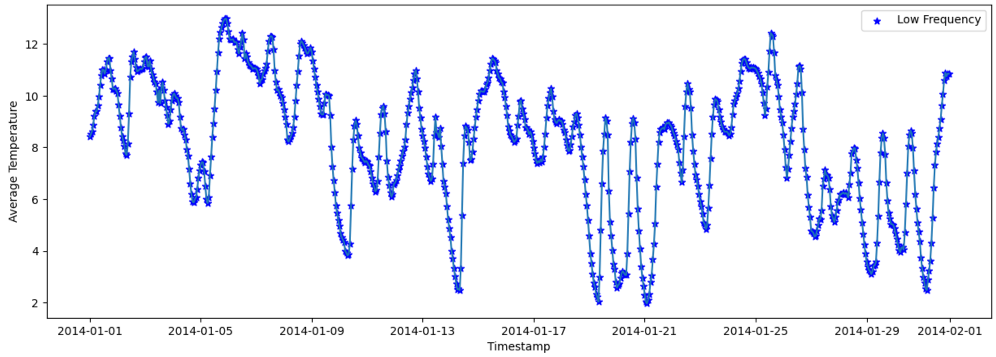


Figure 6: Average temperature. Markers' colours depend on the majority class.

The majority class present in the graph for each timestamp is low frequency.

Question 7

The previous graph G only uses spatial information. To take into account the temporal dynamic, we construct a larger graph H as follows: a node is now *a station at a particular time* and is connected to neighbouring stations (with respect to G) and to itself at the previous timestamp and the following timestamp. Notice that the new spatio-temporal graph H is the Cartesian product of the spatial graph G and the temporal graph G' (which is simply a line graph, without loop).

- Express the Laplacian of H using the Laplacian of G and G' (use Kronecker products).
- Express the eigenvalues and eigenvectors of the Laplacian of H using the eigenvalues and eigenvectors of the Laplacian of G and G'.
- Compute the wavelet transform of the temperature signal.
- Classify nodes into low/medium/high frequency and display the same figure as in the previous question.

Answer 7

- The Laplacian of H using the Laplacian of G and G' is expressed as follows:

$$L(H) = L(G) \otimes I_{n'} + I_n \otimes L(G') \text{ ; where } n = |V(G)| \text{ and } n' = |V(G')|$$

- The eigenvalues and eigenvectors of the Laplacian of H using the eigenvalues and eigenvectors of the Laplacians of G and G' are expressed as follows:
  Let u_i be an eigenvector of $L(G)$ and λ_i the associated eigenvalue with $i \in \{1, .., n\}$ and let u'_j be an eigenvector of $L(G')$ and λ'_j the associated eigenvalue with $j \in \{1, .., n'\}$.

$$\begin{aligned} L(H)(u_i \otimes u'_j) &= (L(G) \otimes I_{n'} + I_n \otimes L(G'))(u_i \otimes u'_j) \\ &= (L(G) \otimes I_{n'})(u_i \otimes u'_j) + (I_n \otimes L(G'))(u_i \otimes u'_j) \\ &= L(G)u_i \otimes I_{n'}u'_j + I_n u_i \otimes L(G')u'_j \\ &= \lambda_i u_i \otimes u'_j + \lambda'_j u_i \otimes u'_j \\ &= (\lambda_i + \lambda'_j)(u_i \otimes u'_j) \end{aligned}$$

  Thus $L(H)$ has $n + n'$ eigenvalues : $\{\lambda_1 + \lambda'_1, \lambda_1 + \lambda'_2, ..., \lambda_n + \lambda'_{n'}\}$ and $n + n'$ eigenvectors : $\{u_1 \otimes u'_1, u_1 \otimes u'_2, ...u_n \otimes u'_{n'}\}$.

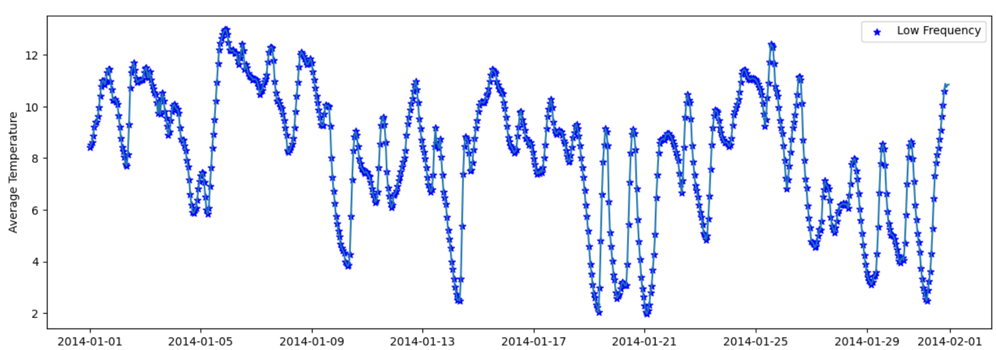


Figure 7: Average temperature. Markers' colours depend on the majority class.

Even when taking into account the temporal dynamic, the majority class present in the graph for each timestamp is low frequency.

References

Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020. ISSN 0165-1684. doi: https://doi.org/10.1016/j.sigpro.2019.107299.

Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48 3:443–53, 1970.