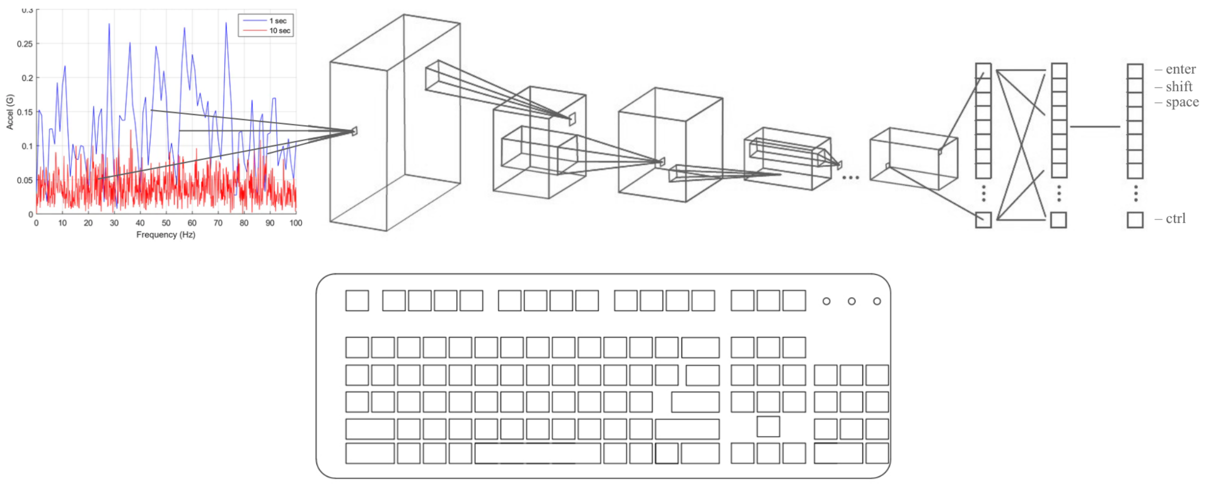
Kingdom of Morocco
Abdelmalek Essaâdi University
Faculty of Sciences and Techniques
Tangier
Department of Electrical Engineering

Signal Processing Project

Department of Electrical Engineering

Study and implementation of an acquisition chain for acoustic analysis of keystrokes and their classification by AI



Realized by:
Oussama AK-HAIL
Assia KARMOUT

2nd Year GEMI

Academic Year: 2024–2025

Contents

Introduction:

With the rapid development of computer technologies, information security has become a major issue. Passwords are widely used to protect access to computer systems, but they remain vulnerable to certain indirect attacks, notably **acoustic side-channel attacks**. Indeed, during typing, involuntary physical information, such as vibrations and sound or mechanical signals, can be emitted and exploited to deduce the actions performed by the user.

In this context, this project focuses on the study of signals generated when typing on a keyboard. The objective is not to hack or steal passwords, but to understand and analyze the security risks related to physical keyboard emissions, in order to raise awareness of potential vulnerabilities and propose adapted protection solutions.

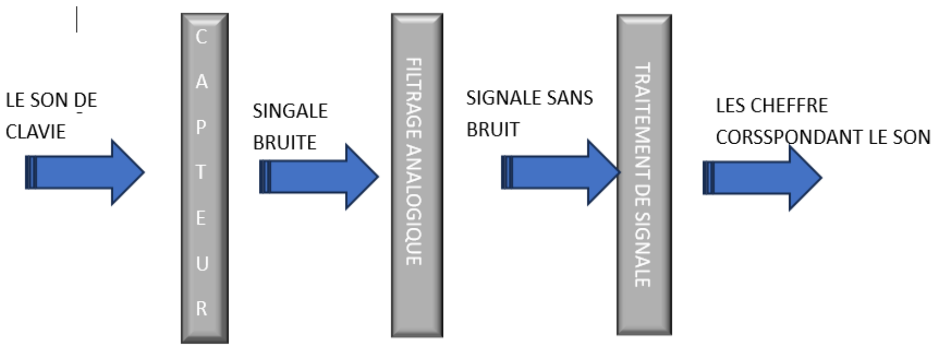
The project involves using an appropriate sensor, such as a vibration sensor or an accelerometer, to capture the signals produced when keys are pressed. These signals are then processed and analyzed to identify distinctive characteristics associated with keystrokes. The results obtained allow for evaluating the system's vulnerability level and better understanding the threats linked to this type of indirect attack.

This work combines concepts of cybersecurity, signal processing, and electronics. It highlights the importance of taking physical emissions into account in the design of secure systems and contributes to reinforcing awareness of good data protection practices.

Chapter 1:

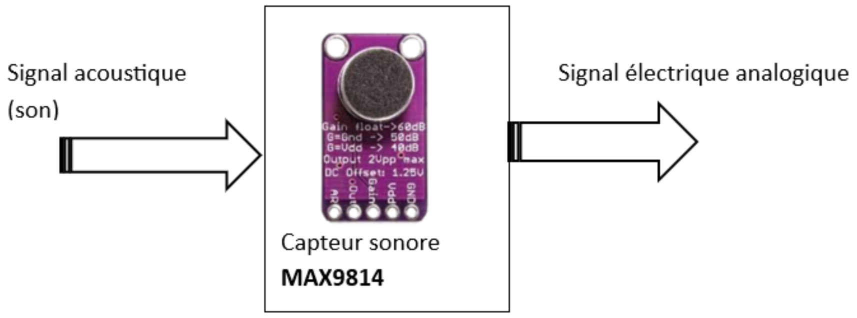
Choice of components to realize a microphone for capturing signals

In this part, we present the choice of components used for the realization of our project. The correct choice of components is an essential step, as it directly influences the performance, reliability, and cost of the system. Each component was selected based on the project's needs, technical constraints, and ease of integration. The main objective is to ensure correct acquisition of keyboard keystroke signals, their effective processing, as well as a clear display of the results. The chosen components allow for capturing signals, conditioning them, and processing them to obtain a functional system adapted to the study of vulnerabilities related to keystrokes.



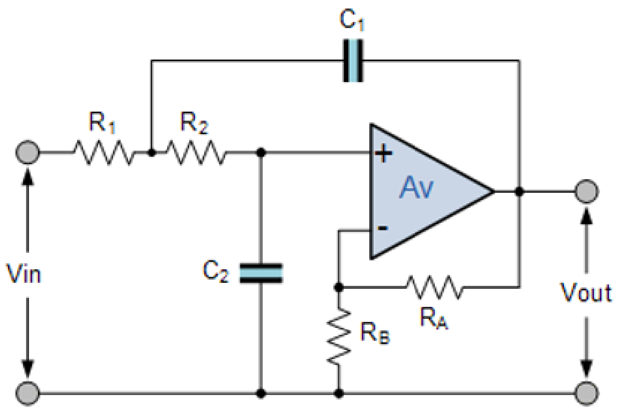
1 Sound Sensor:

In this project, an electret microphone with an amplifier module was chosen, such as the MAX9814, KY-038, or MAX4466 modules. This type of sensor has good sensitivity to weak sounds, especially those produced by keyboard keystrokes. The integrated amplifier module increases the output signal amplitude, making it directly usable by the processing system. Furthermore, these modules are simple to use, inexpensive, and widely used in sound signal processing applications.



2 ANALOG FILTER:

The MAX9814 sensor captures sounds from the external environment, such as keyboard keystrokes as well as ambient background noise. However, the recovered audio signal contains both the useful signal and unwanted noise, mainly at high frequencies. To improve signal quality and eliminate these disturbances, it is necessary to implement a low-pass filter at the sensor output. This filter preserves the frequencies corresponding to the useful sound while attenuating noise components, thus ensuring a cleaner signal usable by the processing system. For our project, we used a Butterworth low-pass filter because the filter does not admit oscillations in the passband or the stopband. To realize this filter, we use this circuit:



Avec : $R_A = 10\,k\Omega\,; R_B = 5.6\,k\Omega\,; R_1 = R_2 = R\,; C_1 = C_2 = C$

$$\boxed{fc=1/2\pi RC}$$

We have an fmax of the input signals at 1000 Hz, so fc = 1500 Hz. Therefore, R = 0.1 K and C = 1.06 nf. Example:
When a person enters a password in a noisy environment, it becomes difficult to distinguish the sound of keystrokes. The signal delivered by the microphone is then affected by ambient noise. This signal is represented in the following figure:

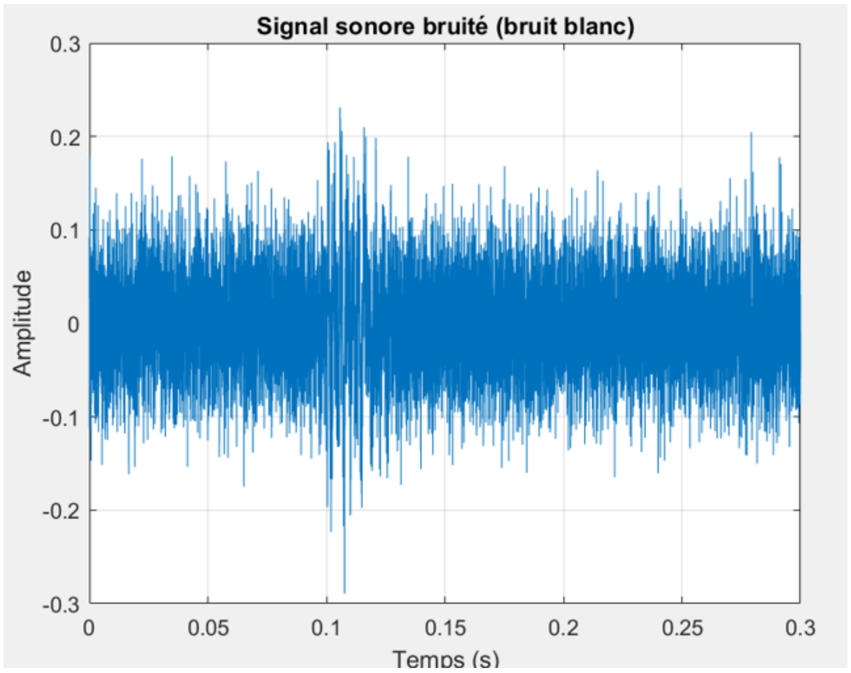


Figure 1: Noisy signal

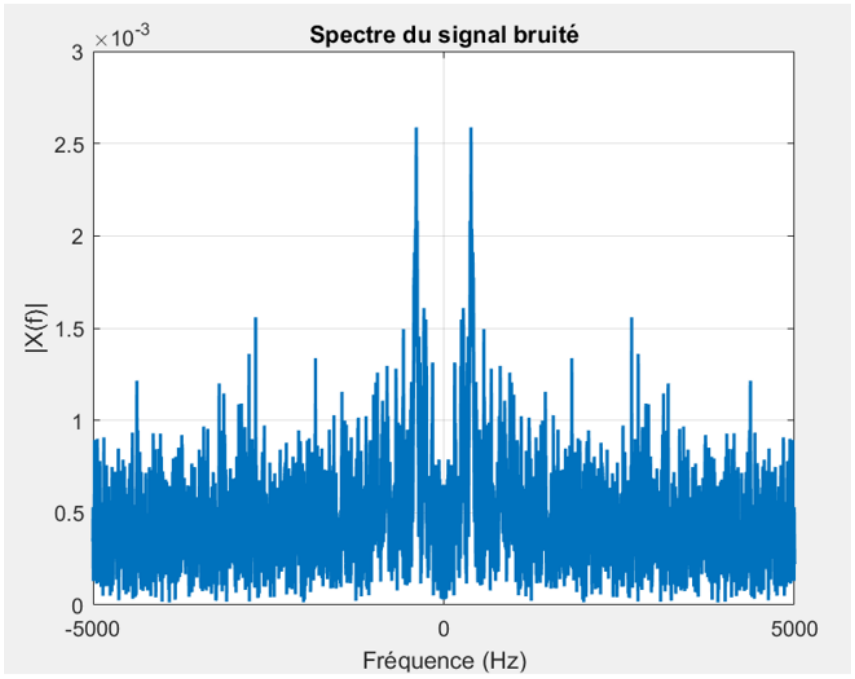


Figure 2: Noisy spectrum

We can observe that this signal is noisy. To eliminate the noise, we use an order 8 Butterworth low-pass filter:

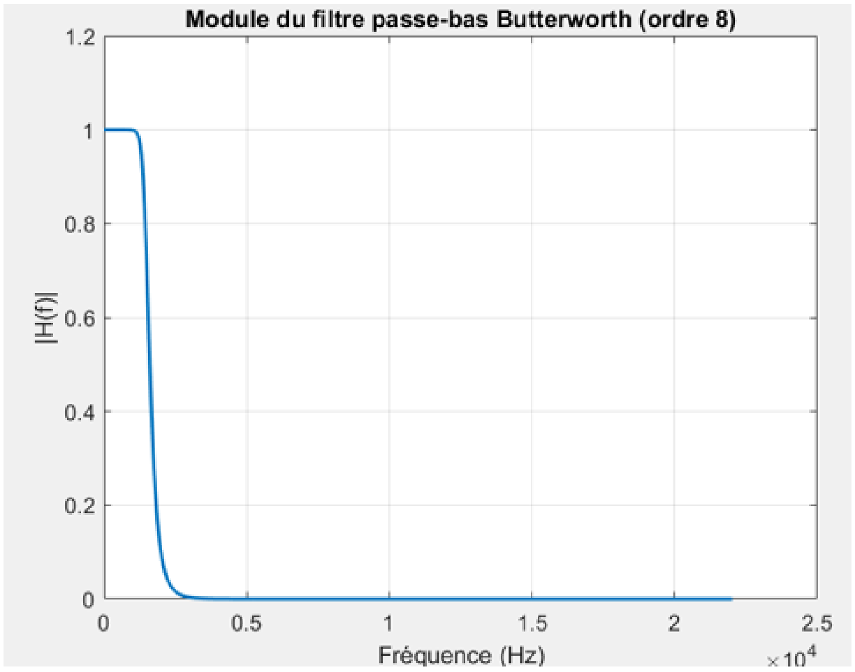
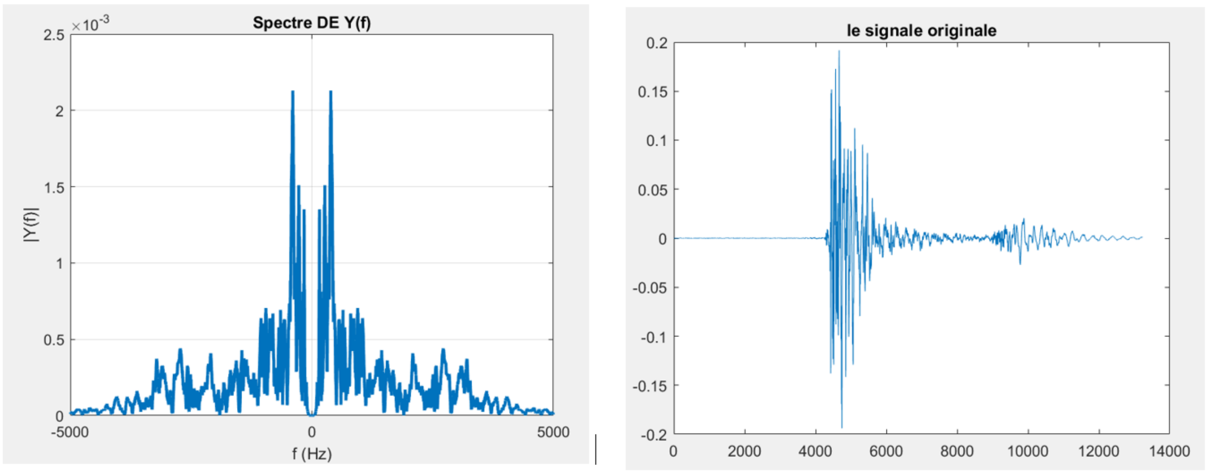


Figure 3: Butterworth low-pass filter

After filtering, we were able to correctly reconstruct the sound of the keystrokes. This signal is represented in the following figure:



Chapter 2:

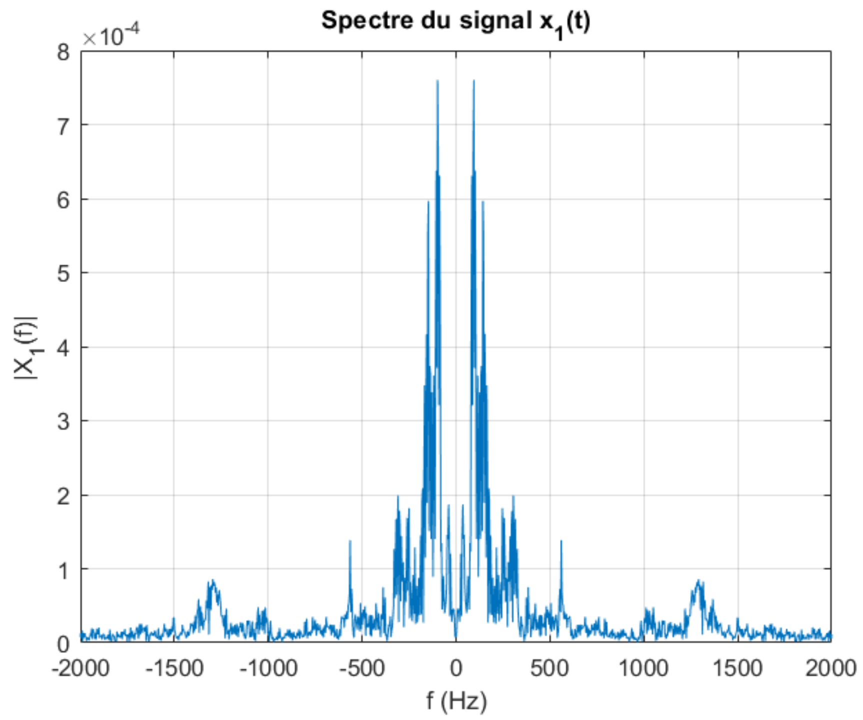
Spectral Extraction and AI Classification

3 Introduction:

After realizing the microphone, the next step consists of collecting the sounds generated by the keystrokes. After collecting the sounds, we use the **Fourier Transform**, specifically the Fast Fourier Transform (FFT), to extract the spectral representation of the keystrokes. However, the obtained spectra cannot be directly used for classification or prediction. It is therefore necessary to resort to a classification method based on artificial intelligence, in particular the **SVM** (Support Vector Machines) method.
Due to the time constraint (one week), the study focused only on two keyboard keys, namely the Space key and the Right Shift key, to simplify the training of the machine learning model.

4 Spectral Extraction:

To illustrate the process, a sample is selected. After the transformation, the results are represented by the graphs below.



The spectrum of a single click on the Space key

We observe that most of the information is located in the frequency range between 0 and 1500 Hz.

5 Artificial Intelligence:

We now have the spectral representation of a keystroke. However, it is impossible to directly determine if this representation corresponds to the Space key or the Right Shift key. Indeed, the spectra are very similar and cannot be differentiated by the naked eye.

This is where artificial intelligence comes in: the Fourier transform provides only frequency and amplitude values, and the SVM allows creating a model capable of classifying these representations from a sufficiently large dataset.

5.1 Dataset Constitution and Preparation

Regarding the dataset, we found an online database containing six samples for each key. In our case, this corresponds to six samples for the Space key and six samples for the Right Shift key. This limited availability forced us to create our own dataset manually. To do this, the microphone was placed close to each key, and each key was struck multiple times to build a dataset large enough to train an accurate model.
Next, we record and proceed with multiple keystrokes. Each keystroke must now be segmented. In this way, we can obtain the Fourier transform of each keystroke, rather than the complete signal which contains a large number of keystrokes. That is why we created a small code that performs this step for us. It automatically detects peaks and cuts the sound before and after these peaks to isolate a single click.

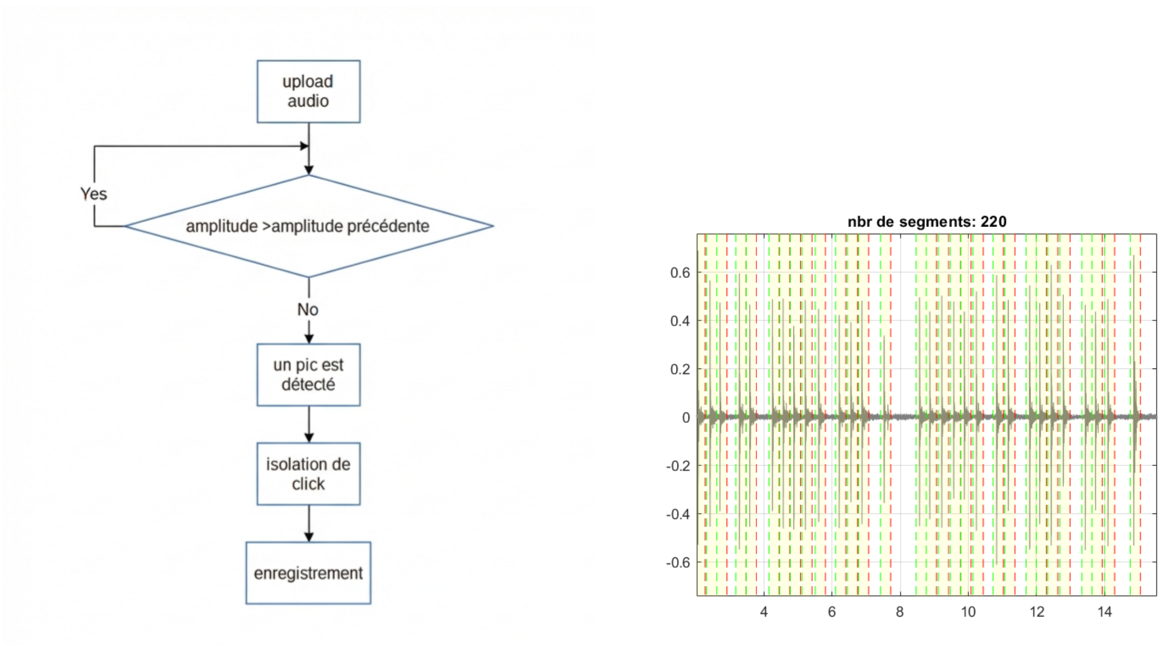
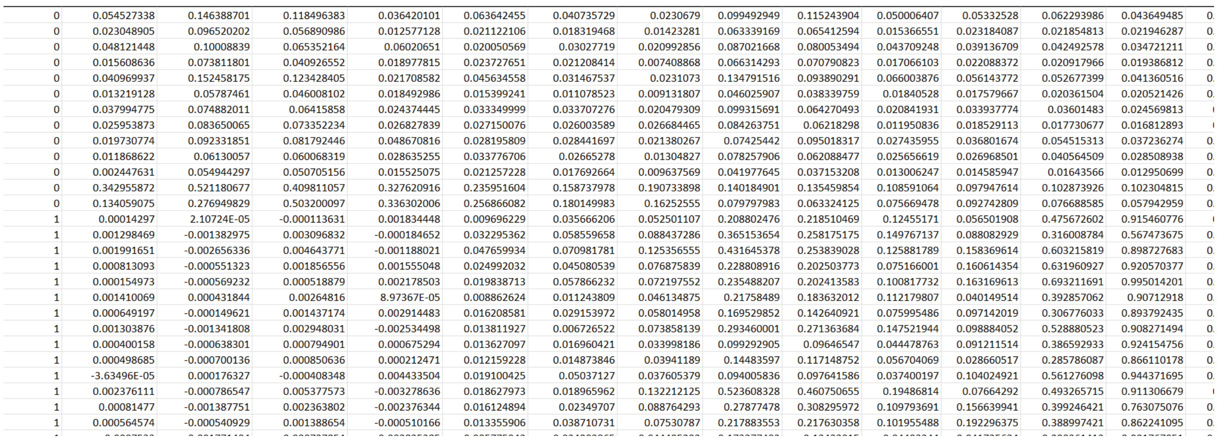


Figure 4: Principle flowchart and segmented plot

We have now obtained a dataset comprising 219 samples of the Right Shift key and 214 samples of the Space key ready to be transformed with the Fourier transform, also associated with a corresponding label, where 1 represents the Space key and 0 represents the Right Shift key.
Finally, our dataset is in the form of a 435x1000 matrix, where the first column indicates the key type, and the other columns represent the values of each frequency of the corresponding spectrum.

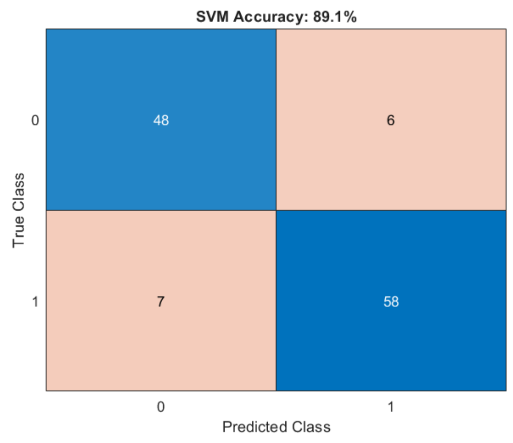


0	0.054527338	0.146388701	0.118496383	0.036420101	0.063642455	0.040735729	0.0230679	0.099492949	0.115243904	0.050006407	0.05332528	0.062293986	0.043649485	0.
0	0.023048905	0.096520202	0.056890986	0.012577128	0.021122106	0.018319468	0.01423281	0.063339169	0.065412594	0.015366551	0.023184087	0.021854813	0.021946287	0.
0	0.048121448	0.10008839	0.065352164	0.06020651	0.020050569	0.03027719	0.020992856	0.087021668	0.080053494	0.043709248	0.039136709	0.042492578	0.034721211	0.
0	0.015608636	0.073811801	0.040926552	0.018977815	0.023727651	0.021208414	0.007408868	0.066314293	0.070790823	0.017066103	0.022088372	0.020917966	0.019386812	0.
0	0.040969937	0.152458175	0.123428405	0.021708582	0.045634558	0.031467537	0.0231073	0.134791516	0.093890291	0.066003876	0.056143772	0.052677399	0.041360516	0.
0	0.013219128	0.05787461	0.046008102	0.018492986	0.015399241	0.011078523	0.009131807	0.046025907	0.038339759	0.01840528	0.017579667	0.020361504	0.020521426	0.
0	0.037994775	0.074882011	0.06415858	0.024374445	0.033349999	0.033707276	0.020479309	0.099315691	0.064270493	0.020841931	0.033937774	0.03601483	0.024569813	
0	0.025953873	0.083650065	0.073352234	0.026827839	0.027150076	0.026003589	0.026684465	0.084263751	0.06218298	0.011950836	0.018529113	0.017730677	0.016812893	
0	0.019730774	0.092331851	0.081792446	0.048670816	0.028195809	0.028441697	0.021380267	0.07425442	0.095018317	0.027435955	0.036801674	0.054515313	0.037236274	0.
0	0.011868622	0.06130057	0.060068319	0.028635255	0.033776706	0.02665278	0.01304827	0.078257906	0.062088477	0.025656619	0.026968501	0.040564509	0.028508938	0.
0	0.002447631	0.054944297	0.050705156	0.015525075	0.021257228	0.017692664	0.009637569	0.041977645	0.037153208	0.013006247	0.014585947	0.01643566	0.012950699	0.
0	0.342955872	0.521180677	0.409811057	0.327620916	0.235951604	0.158737978	0.190733898	0.140184901	0.135459854	0.108591064	0.097947614	0.102873926	0.102304815	0.
0	0.134059075	0.276949829	0.503200097	0.336302006	0.256866082	0.180149983	0.16252555	0.079797983	0.063324125	0.075669478	0.092742809	0.076688585	0.057942959	0.
1	0.00014297	2.10724E-05	-0.000113631	0.001834448	0.009696229	0.035666206	0.052501107	0.208802476	0.218510469	0.12455171	0.056501908	0.475672602	0.915460776	
1	0.001298469	-0.001382975	0.003096832	-0.000184652	0.032295362	0.058559658	0.088437286	0.365153654	0.258175175	0.149767137	0.088082929	0.316008784	0.567473675	0.
1	0.001991651	-0.002656336	0.004643771	-0.001188021	0.047659934	0.070981781	0.125356555	0.431645378	0.253839028	0.125881789	0.158369614	0.603215819	0.898727683	0.
1	0.000813093	-0.000551323	0.001856556	0.001555048	0.024992032	0.045080539	0.076875839	0.228808916	0.202503773	0.075166001	0.160614354	0.631960927	0.920570377	0.
1	0.000154973	-0.000569232	0.000518879	0.002178503	0.019838713	0.057866232	0.072197552	0.235488207	0.202413583	0.100817732	0.163169613	0.693211691	0.995014201	0.
1	0.001410069	0.000431844	0.00264816	8.97367E-05	0.008862624	0.011243809	0.046134875	0.21758489	0.183632012	0.112179807	0.040149514	0.392857062	0.90712918	0.
1	0.000649197	-0.000149621	0.001437174	0.002914483	0.016208581	0.029153972	0.058014958	0.169529852	0.142640921	0.075995486	0.097142019	0.306776033	0.893792435	0.
1	0.001303876	-0.001341808	0.002948031	-0.002534498	0.013811927	0.006726522	0.073858139	0.293460001	0.271363684	0.147521944	0.098884052	0.528880523	0.908271494	0.
1	0.000400158	-0.000638301	0.000794901	0.000675294	0.013627097	0.016960421	0.033998186	0.099292905	0.09646547	0.044478763	0.091211514	0.386592933	0.924154756	0.
1	0.000498685	-0.000700136	0.000850636	0.000212471	0.012159228	0.014873846	0.03941189	0.14483597	0.117148752	0.056704069	0.028660517	0.285786087	0.866110178	0.
1	-3.63496E-05	0.000176327	-0.000408348	0.004433504	0.019100425	0.05037127	0.037605379	0.094005836	0.097641586	0.037400197	0.104024921	0.561276098	0.944371695	0.
1	0.002376111	-0.000786547	0.005377573	-0.003278636	0.018627973	0.018965962	0.132212125	0.523608328	0.460750655	0.19486814	0.07664292	0.493265715	0.911306679	
1	0.00081477	-0.001387751	0.002363802	-0.002376344	0.016124894	0.02349707	0.088764293	0.27877478	0.308295972	0.109793691	0.156639941	0.399246421	0.763075076	0.
1	0.000564574	-0.000540929	0.001388654	-0.000510166	0.013355906	0.038710731	0.07530787	0.217883553	0.217630358	0.101955488	0.192296375	0.388997421	0.862241095	0.

Figure 5: Final dataset

5.2 Training

In this part, we loaded the final dataset containing the Fourier characteristics of the keys and their labels. We shuffled and split the data into a training set (80%) and a test set (20%) to prevent the model from learning the order of examples. For the SVM, we used a linear kernel, standardized the data so that high-energy frequencies do not dominate, and explicitly defined the classes [0,1]



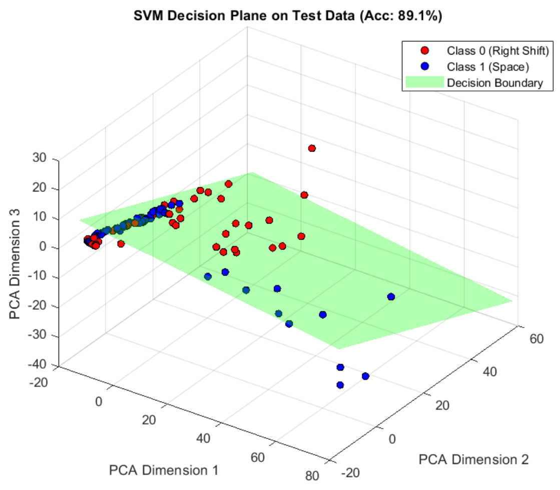


Figure 6: Confusion matrix and 3D visualization of separability

According to the confusion matrix, the accuracy of our model is 89.1

It should be noted that the visualization of separability in our case does not reflect reality, because our matrix has 1000 columns. In other words, it is impossible to visualize it correctly, as the maximum visualization dimension in MATLAB is 3D.

5.3 Prediction of New Signals

This is now the prediction phase. We imported signals that had not yet been processed by our model and applied the Fourier transform. Then, we converted the signal to mono, selected the first 1000 frequencies, and then performed the prediction. The result returns 1 if the signal corresponds to a click on the Space key, and 0 if it corresponds to a click on the Right Shift key.
Here is the illustrative flowchart of the algorithm.

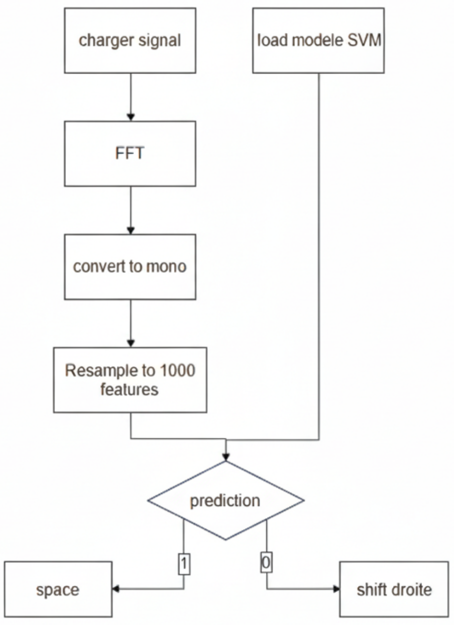


Figure 7: Prediction flowchart

These results correspond to the code outputs. It first displays the input signal converted to mono, then the spectrum of the signal obtained after the Fourier transformation. Then, it presents the model's prediction as well as the confidence level associated with this prediction. The results show that the model works very well overall. In some cases, the confidence rate is relatively low, but the prediction remains correct. In other cases, the confidence level can reach very high values (e.g., 100

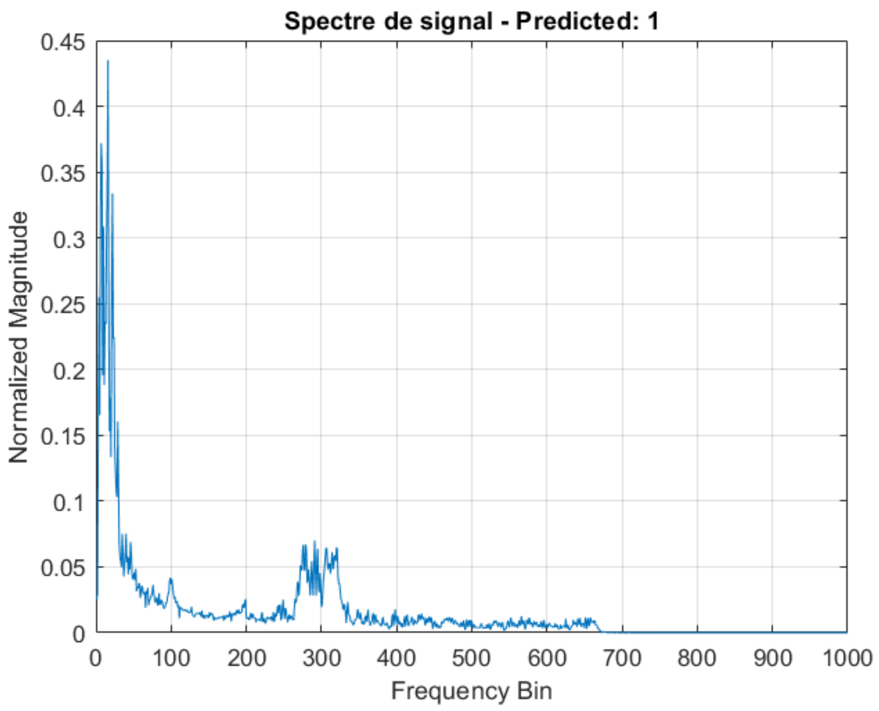
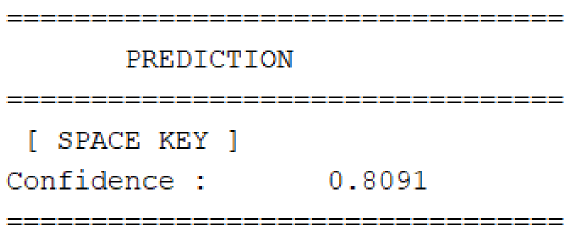


Figure 8: Signal spectrum



```
==================================
        PREDICTION
==================================
 [ SPACE KEY ]
Confidence :        0.8091
==================================
```

Figure 9: Prediction result

6 Future Versions

For future versions of our project, two important areas for improvement are envisaged. The first consists of using the PC microphone directly. This approach allows for a fixed positioning of the microphone, which improves precision in distinguishing between different keys. The second consists of modifying the model's learning method, from binary prediction to a **one versus all** approach, to allow the detection and classification of multiple keys.

This method relies on executing multiple binary classifications, given that SVM does not allow predicting more than two classes directly. For example, if we wish to classify a set of keys ranging from 0 to 9, the model successively performs predictions of the type "0 against all", then "1 against all", and so on. Each prediction provides a probability as well as a confidence level, and the key corresponding to the highest probability is considered the detected key.

This approach was the initial idea of the project. However, due to time constraints, it was not possible to implement it up to this level of multi-class classification.

7 How to Protect Against Side-Channel Attacks

As mentioned previously, this report has an educational objective and aims to show how acoustic side-channel attacks can be exploited. It is therefore legitimate to ask how to protect oneself against this type of attack.

Several solutions exist. The simplest is to add white noise to the microphone signal. However, this approach strongly degrades sound quality, which prevents the microphone from correctly fulfilling its main function.

In connection with the first chapter of this project, another solution consists of applying inverse filtering, notably a high-pass filter with a cutoff frequency of around 1500Hz. In this way, the microphone would no longer capture the sounds generated by keyboard keystrokes.

A more realistic alternative consists of performing this filtering directly at the software level, as is generally the case in real applications. This method relies on suppressing the characteristic frequencies of the keyboard, which are mainly located in the low frequencies, below 1500 Hz.

$$\tilde{X}(f) = \frac{1}{2N+1} \sum_{k=f-N}^{f+N} X(k) \tag{1}$$

Finally, another technique called spectral smoothing can be used. It consists of smoothing the peaks of the spectrum, which are key elements for key detection and classification by the SVM, as explained previously. This method thus allows significantly reducing the effectiveness of the attack.

$$Y(f) = \max\left(X(f) - \alpha\, N(f),\, 0\right) \tag{2}$$

Annex:

All codes, data, samples, and the model used in this project have been made available on GitHub. They can be consulted by scanning this QR code

or by clicking on this link:https://github.com/OussamaAKHAIL/signal-and-systems-project