



UNIVERSITÉ CÔTE D'AZUR 

# Approches Deep Learning à la détection d'anomalies dans un système à temps réel

ADAM BOND  
MOHAMED-AMINE ROMDHANE

Encadré par:  
ENRICO FORMENTI



# Table des matières

1. Introduction
2. Simulation & Génération des données
3. Approches Deep Learning
  - a. Approche Naïve avec CNN
  - b. Extraction de features avec TSFresh
  - c. Détection d'anomalies en tant qu'objets
4. Conclusion



# Introduction

**Contexte:** Répondre à un besoin d'une startup qui propose des services d'optimisation de consommation de l'eau dans le milieu agricole

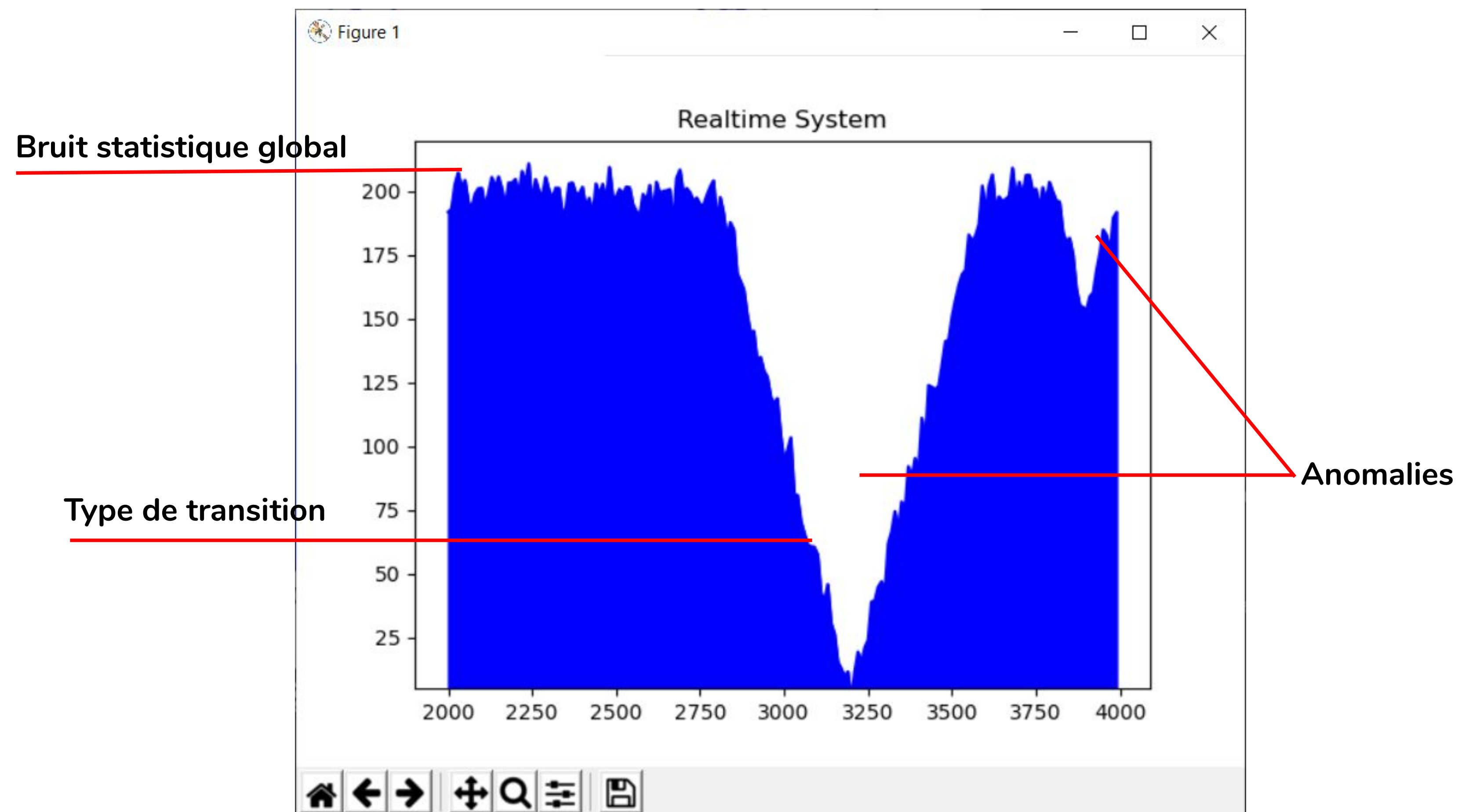
**But:** Détecter les anomalies sur les courbes d'états des senseurs de pression des pompes à eaux à l'aide d'approches deep learning



**Problème:** Comment obtenir un dataset pour entraîner nos modèles ?

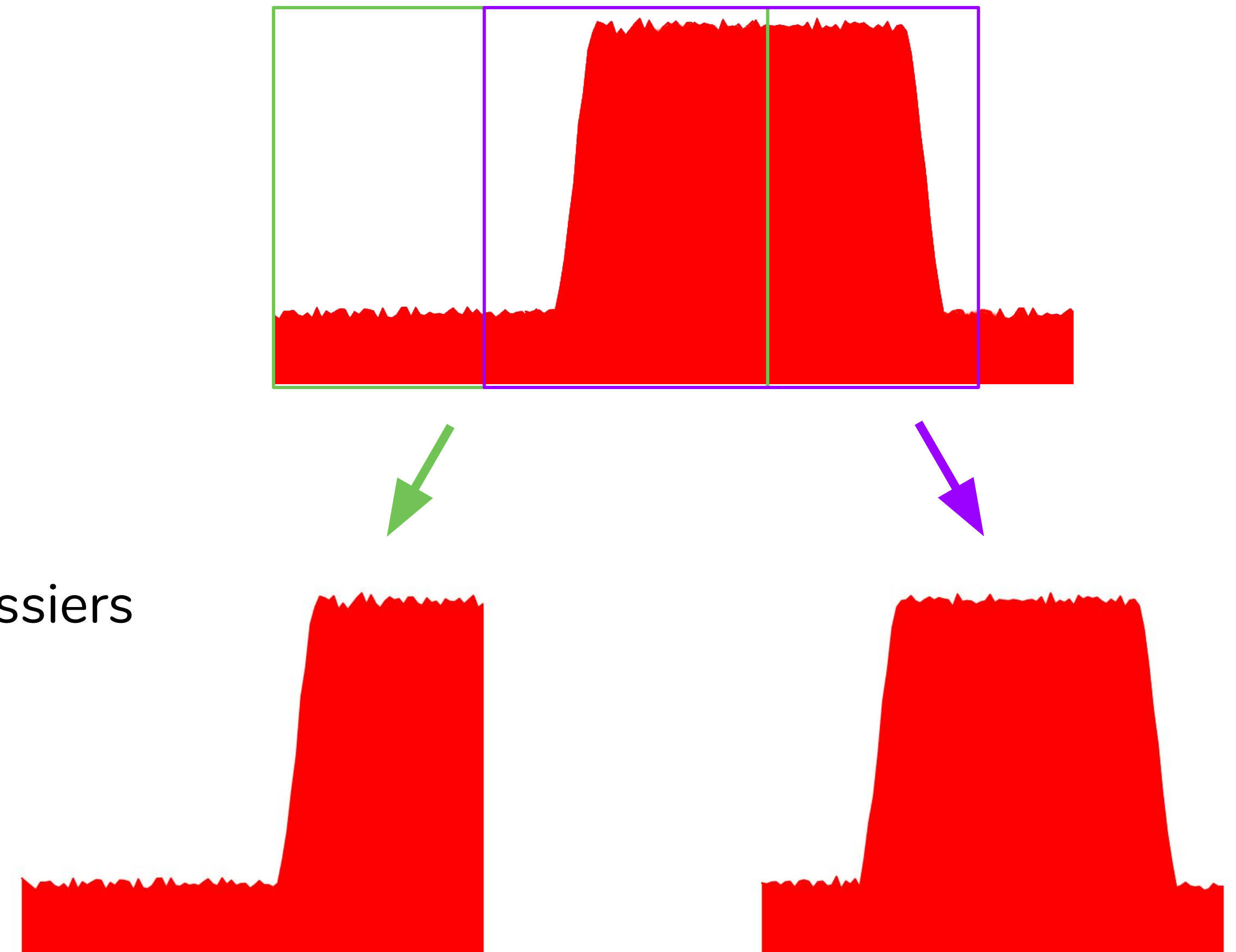
- L'entreprise n'a donné que quelques aperçus confidentiels
- Il faut trouver un moyen de modéliser le problème afin de générer un dataset utilisable
- Solution: Ecriture d'un simulateur modulaire pour générer des séries "stables", ou avec "malfonctions"

# Simulation des données



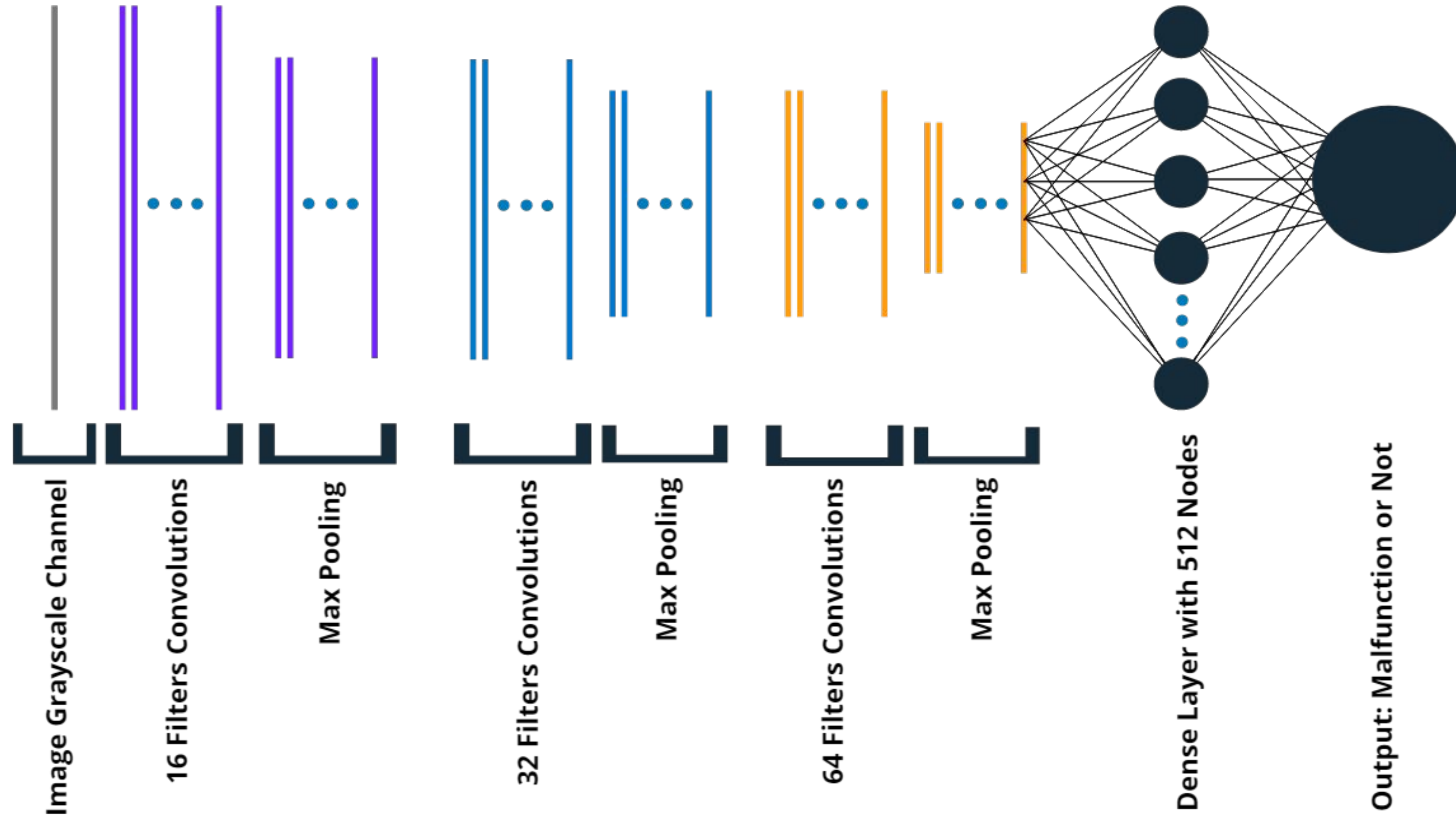
# Génération des données

- On génère un système avec ou sans anomalies
- On enlève les axes, légende, titre, padding, etc...
- On fixe une taille de “fenêtre” (scale), puis on la fait bouger au fur et à mesure qu’on capture la figure
- On sauvegarde la figure en format png dans des dossiers d’entraînement et de tests

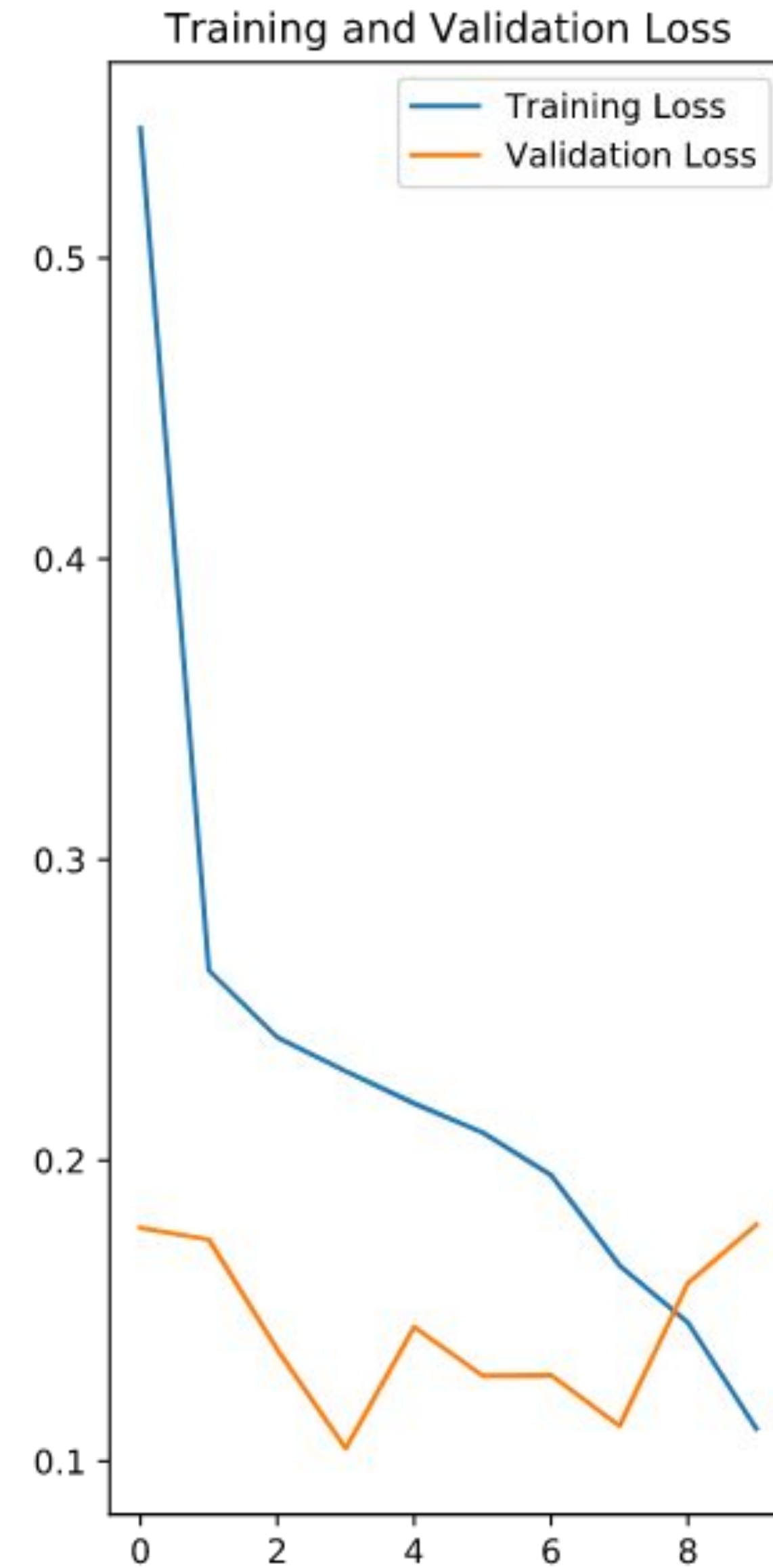
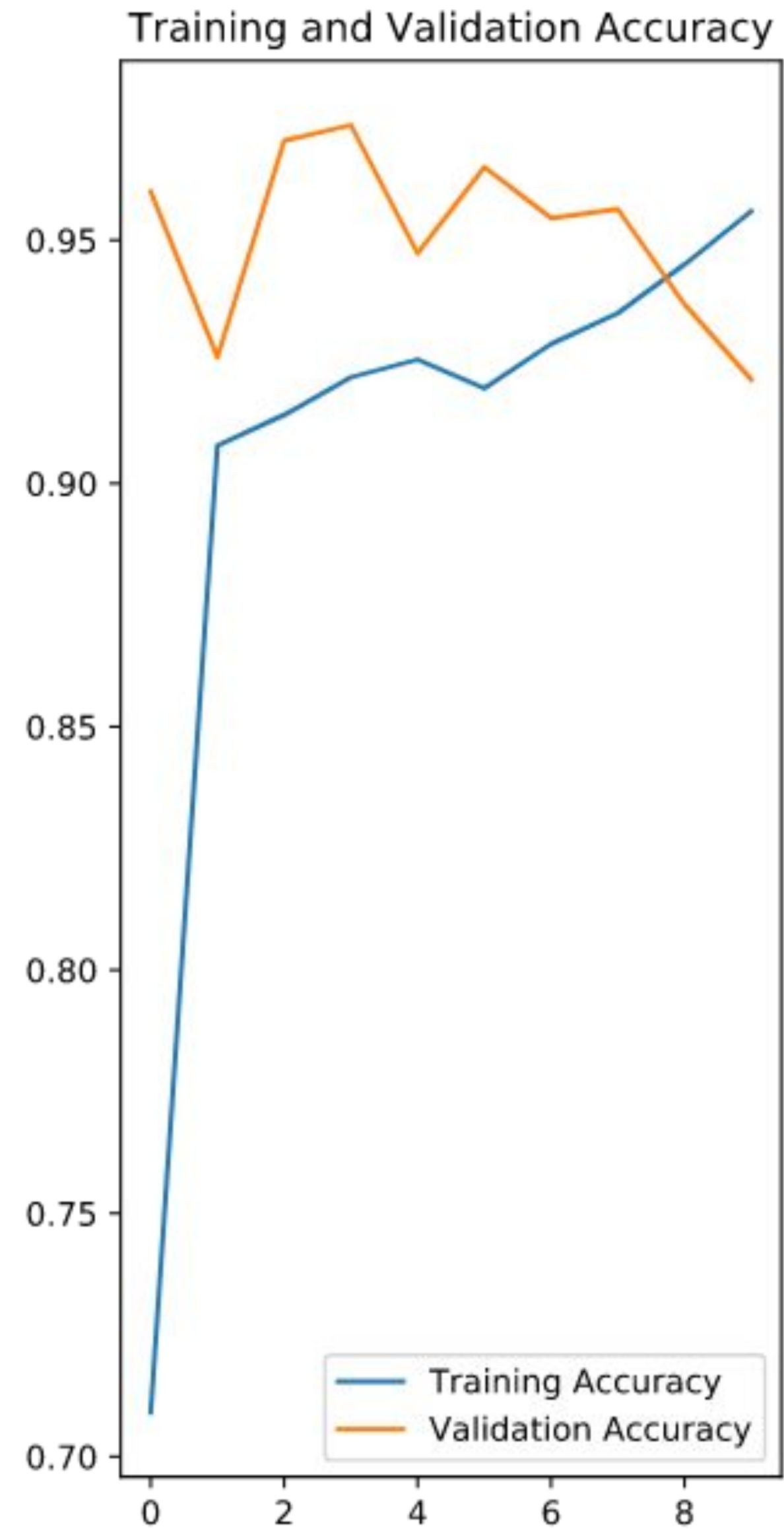




# Approche naïve avec CNN



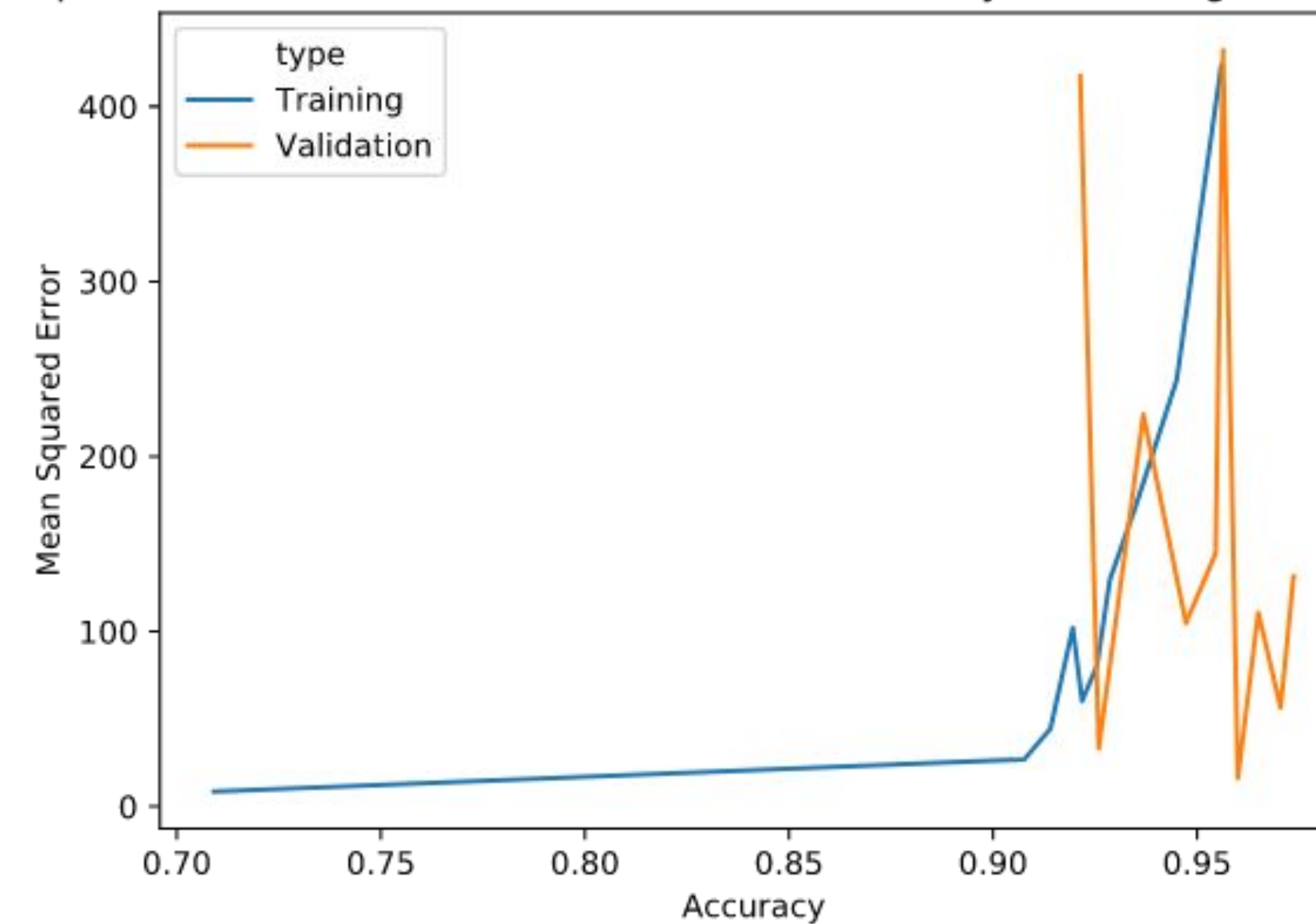
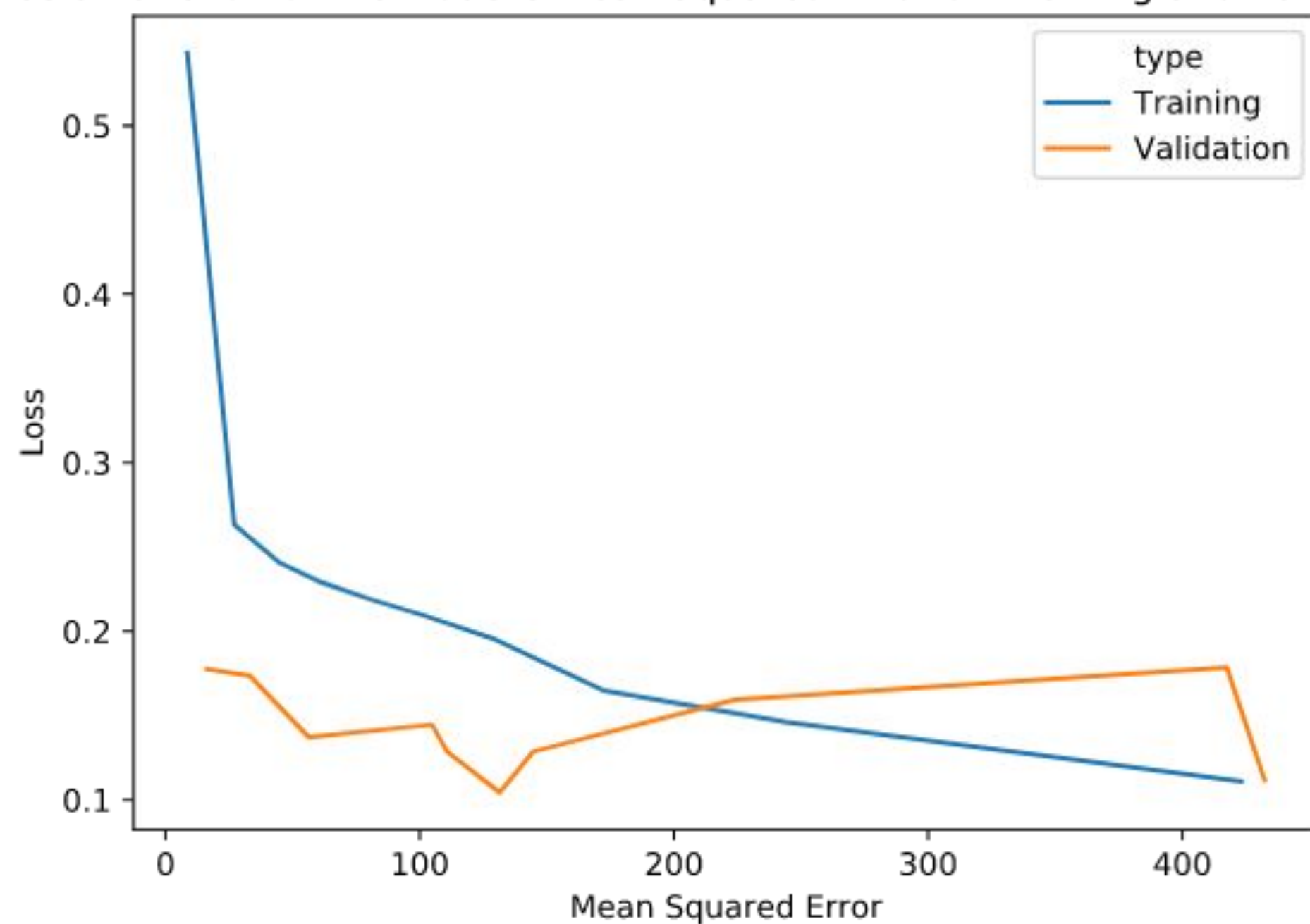
# Approche naïve avec CNN





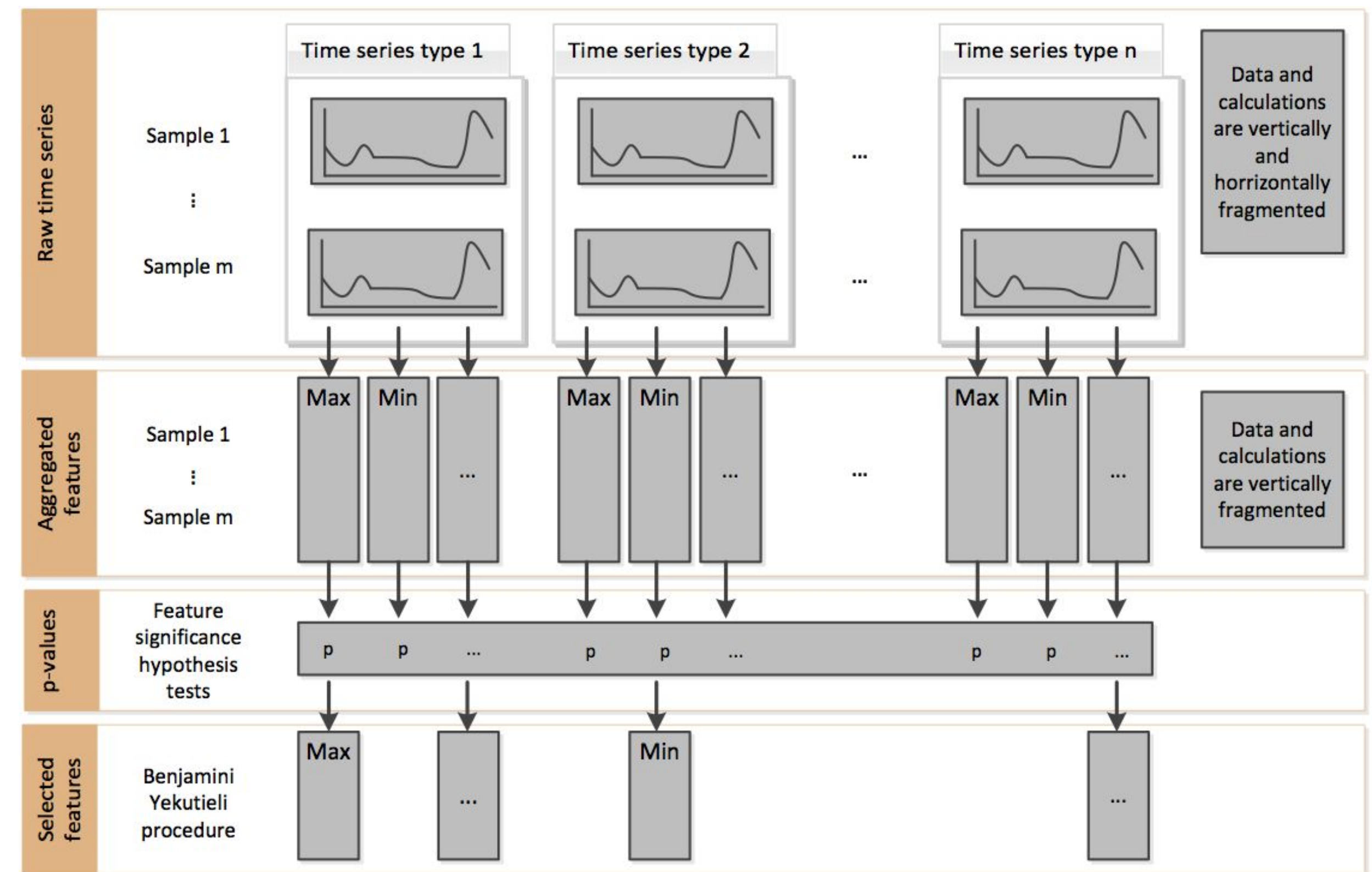
# Approche naïve avec CNN

Loss as a Function of the Model's Mean Squared Error on Training and Validation Sets    Mean Squared Error as a Function of the Model's Accuracy on Training and Validation Sets



# Extraction des features avec TSFresh

- Génération de 1000 séries temporelles
- Calcul de features liés à des séries temporelles (max, min, etc)
- Test du pouvoir prédictif de chaque feature
- Assignation d'une p-value



# Extraction des features avec TSFresh

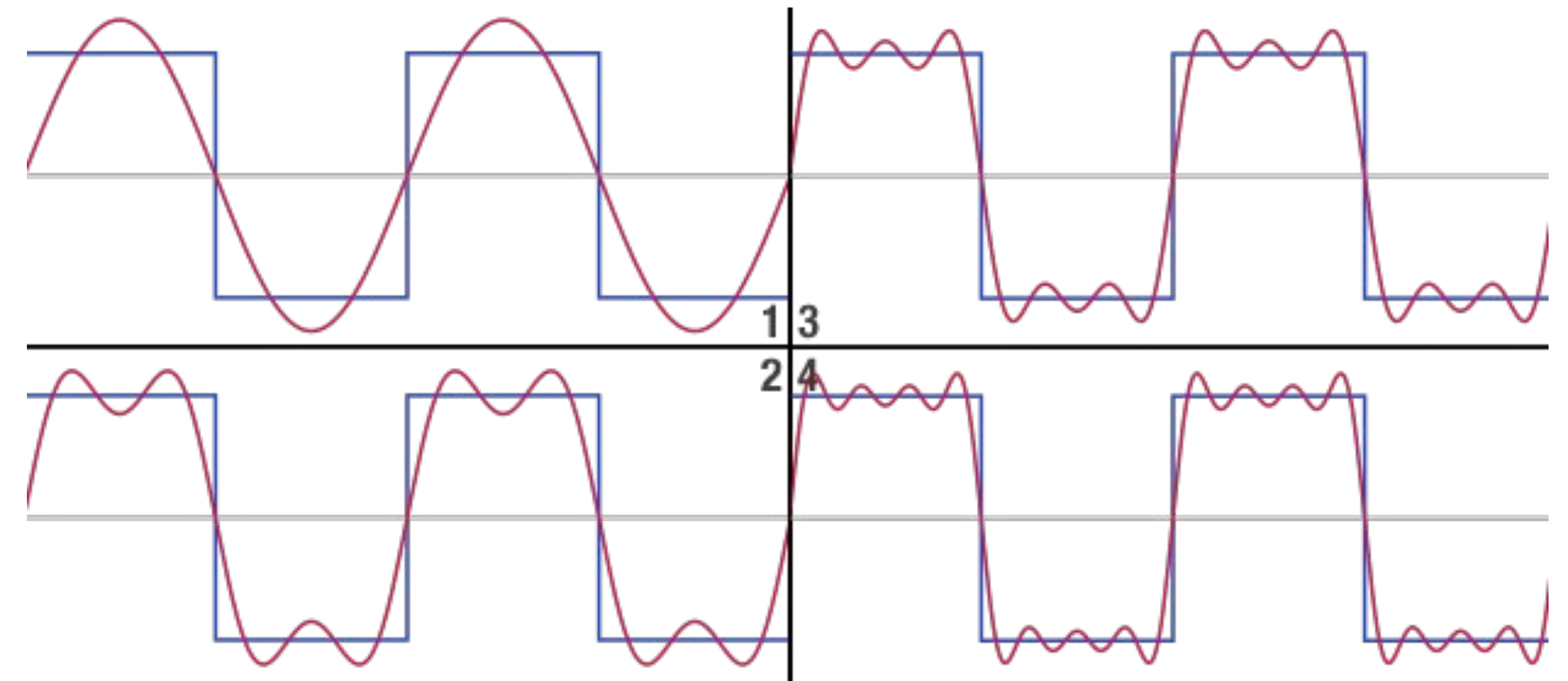
- Résultats: Coefficients de fourier sont de loin les plus efficaces
- Les premiers coefficients sont sur-représentés

Feature	P-Value
FFT Coeff. 1	1.00E-126
FFT Coeff. 2	1.84E-110
FFT Coeff. 3	6.58E-99
FFT Coeff. 5	9.98E-99
FFT Coeff. 4	4.00E-98
FFT Coeff. 7	1.10E-96
FFT Coeff. 6	1.90E-91
FFT Coeff. 23	2.88E-87
FFT Coeff. 11	6.80E-87
FFT Coeff. 17	1.37E-86



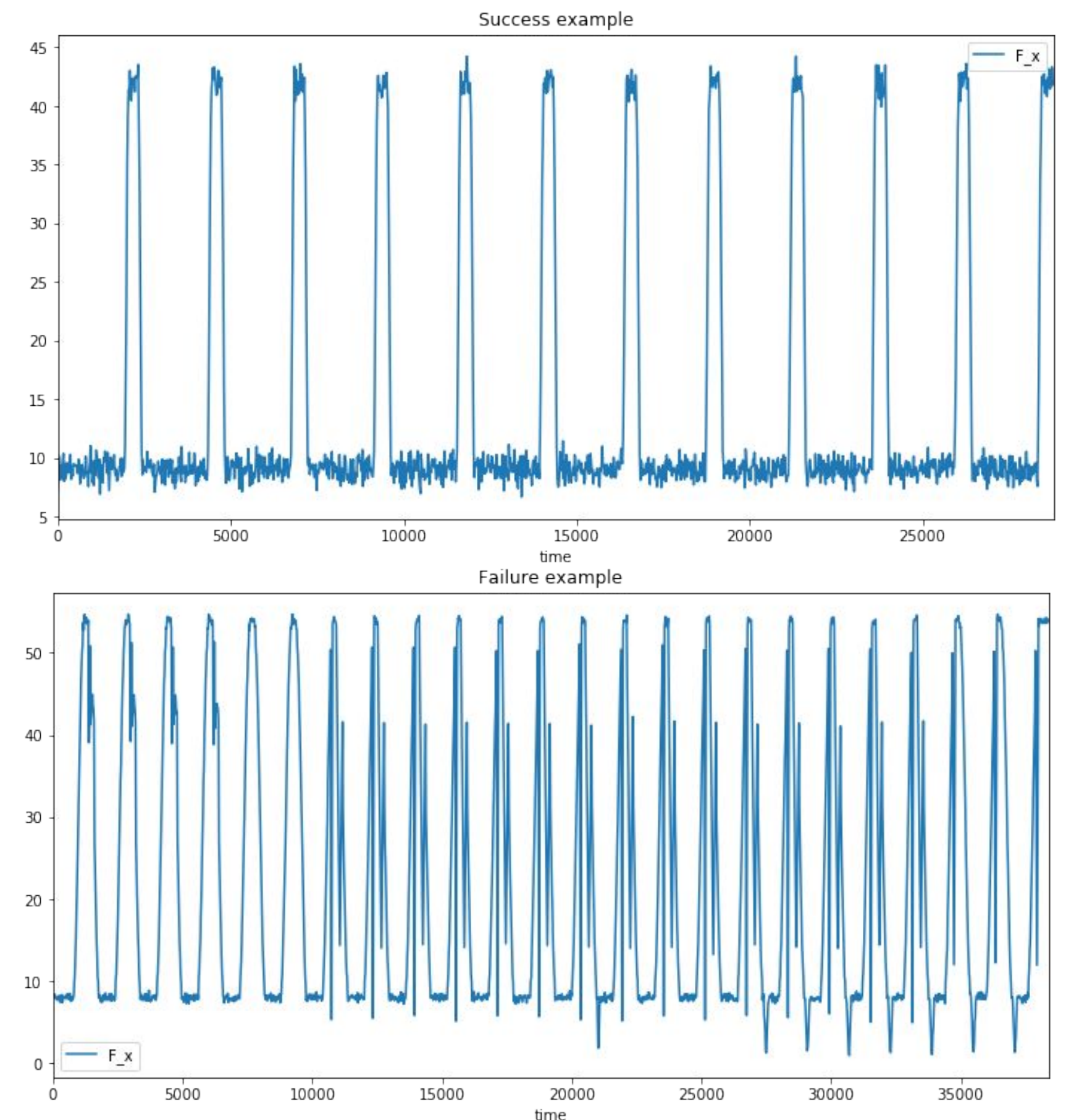
# Extraction des features avec TSFresh

- Une série temporelle stable de notre système ressemble à un signal carré
- Très bien construit avec une addition de sinusoïdes
- Quand il y a des malfunctions, les pics sont plus nombreux -> plus de composantes de hautes fréquences dans le signal



# Extraction des features avec TSFresh

- Les malfunctions sont 'aléatoires' par nature, alors que les séries stables se ressemblent
- Deux phénomènes à haute fréquence: le bruit et les malfunctions
- Plus le nombre de phénomènes augmente, moins les basses fréquences sont utiles





# Extraction des features avec TSFresh

- Arbre de décision entraîné sur les N premières features
- Bonne performance, même avec seulement le premier coefficient de Fourier (>85% précision)
- Malgré un nombre élevé de features disponibles, sans contraintes l'arbre s'arrête à une profondeur basse (<20)



# Détection d'anomalies en tant qu'objets

- On peut considérer une anomalie comme un objet dans l'image
- Utiliser la technique de Deep Learning de détection d'objets pour détecter une anomalie
- Avantage: Détection en temps réel

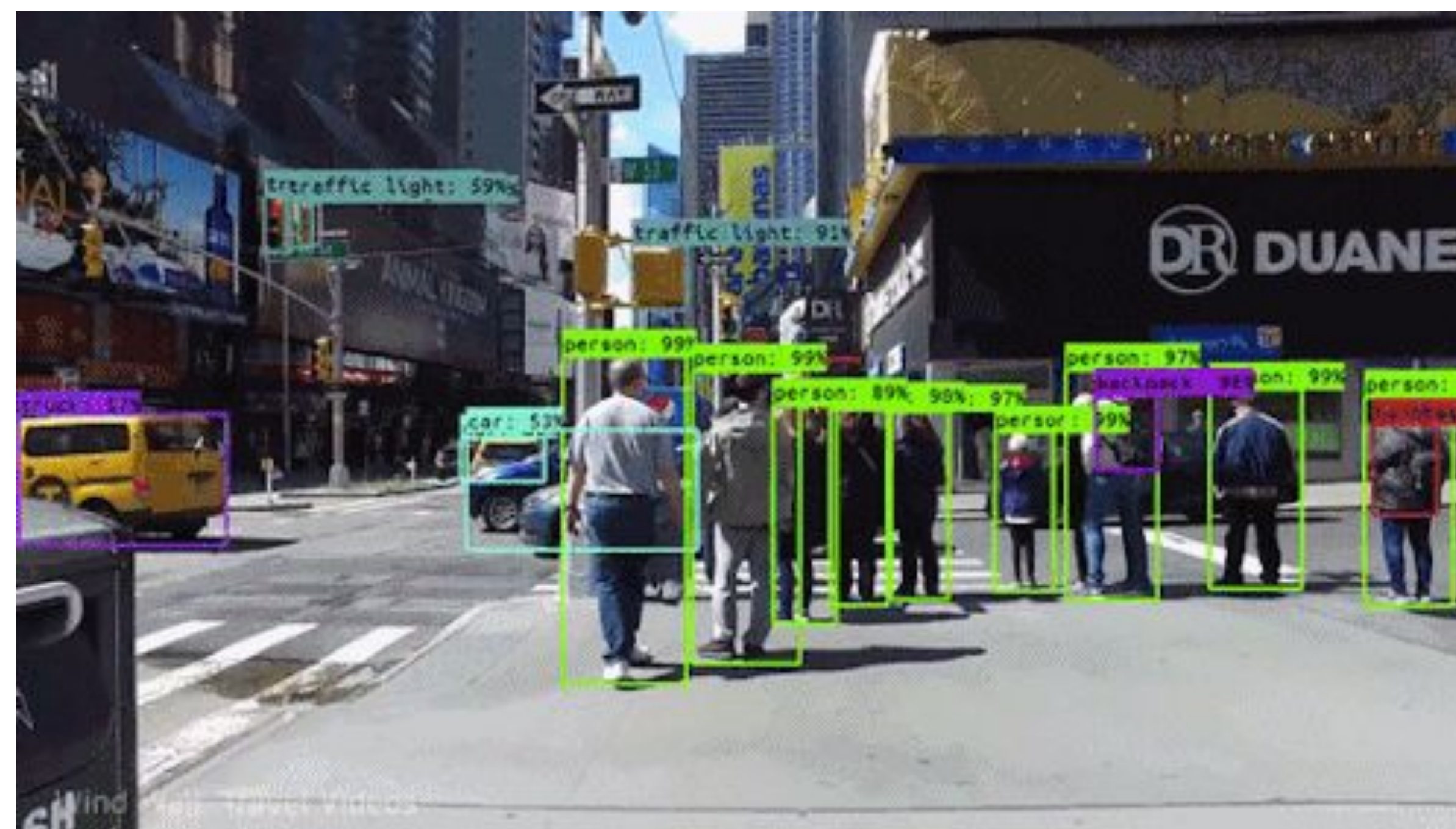
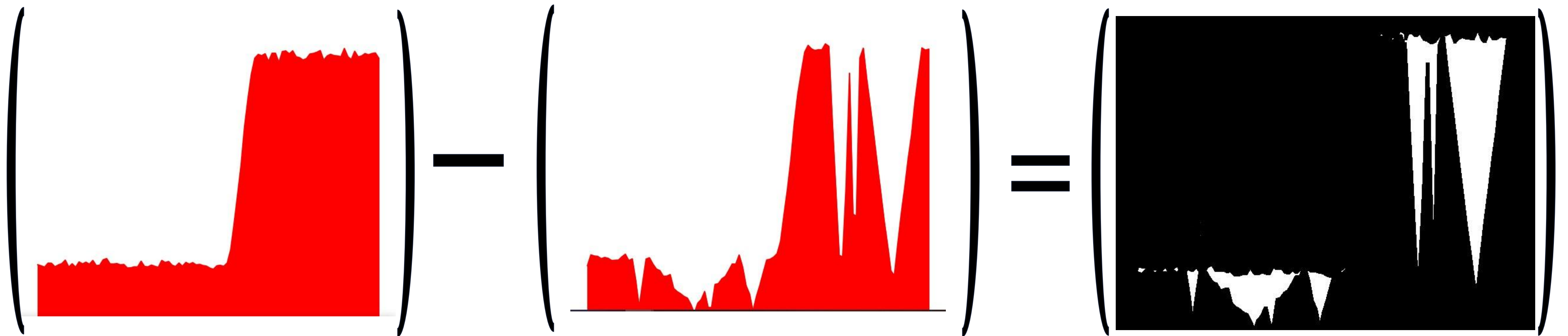


Image prise du repo Detect-Me de abhishek1605 sur GitHub

# Détection d'anomalies en tant qu'objets

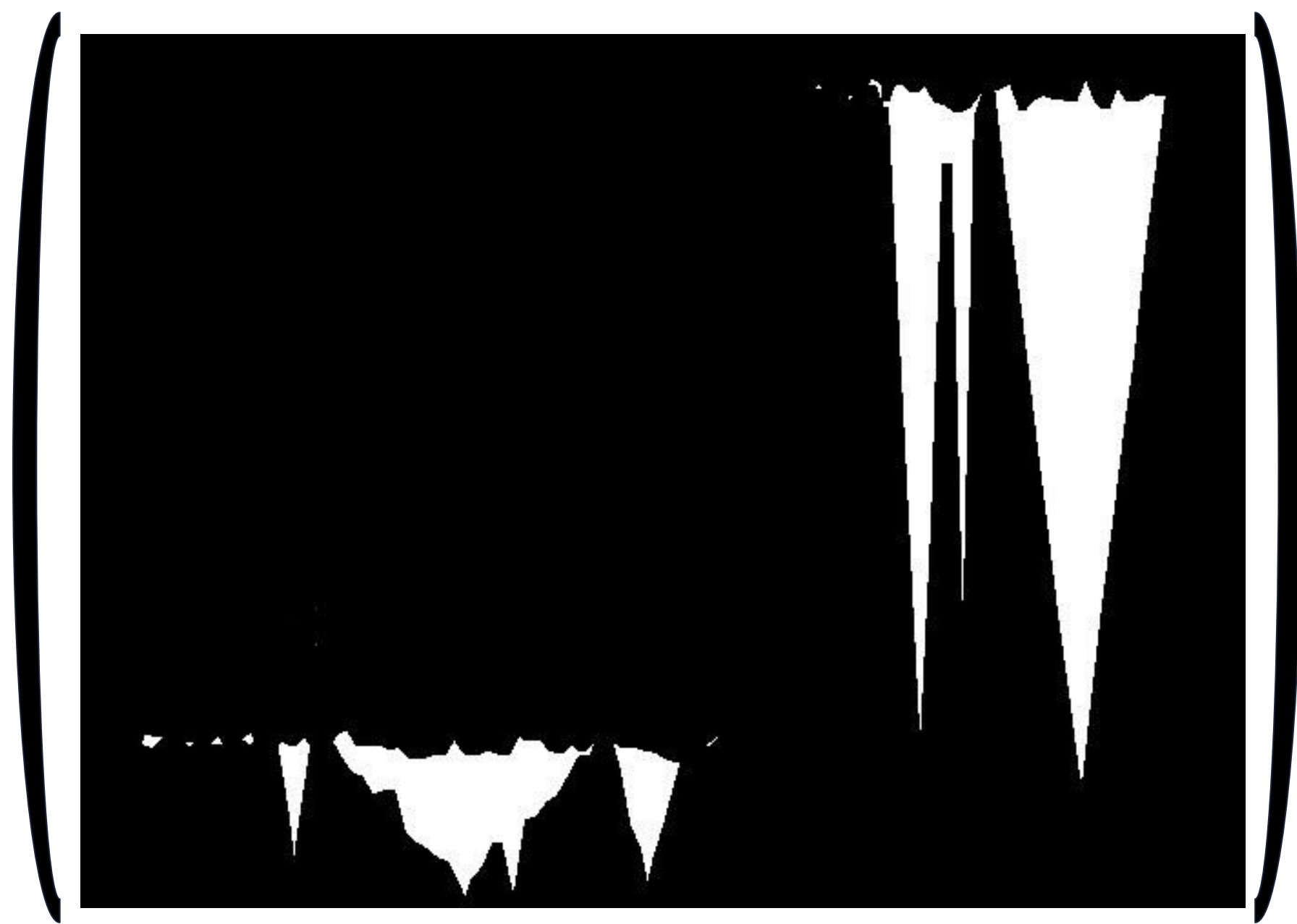
Génération des objets



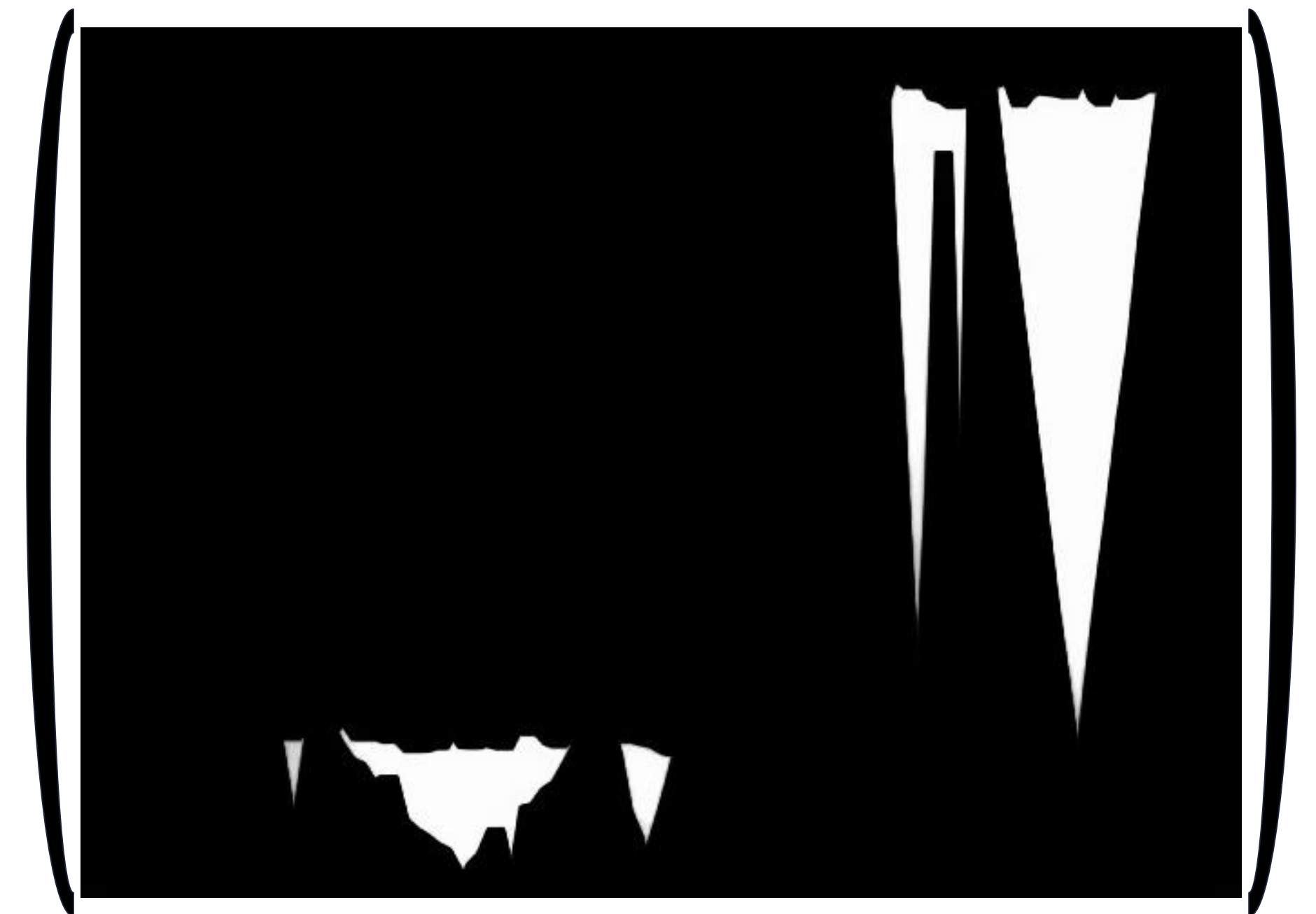


# Détection d'anomalies en tant qu'objets

## Génération des objets

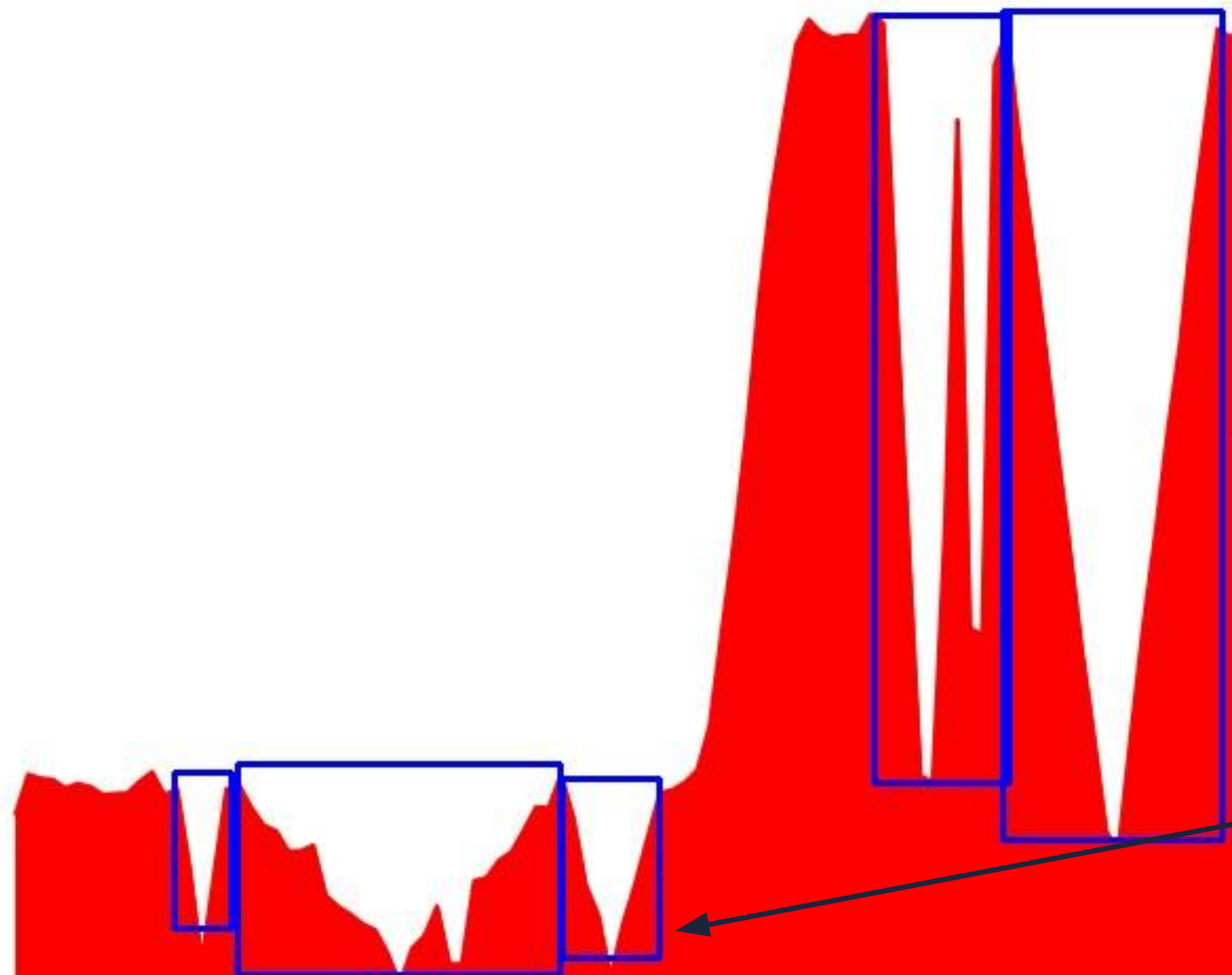


Fonction d'Érosion  
d'OpenCV



# Détection d'anomalies en tant qu'objets

## Génération des objets



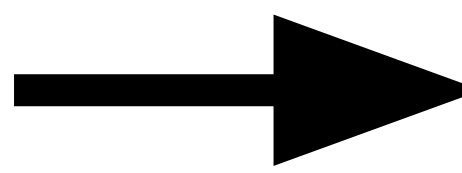
On génère un CSV avec les coordonnées de chaque bounding box et son image associée

**Anomalies isolées**



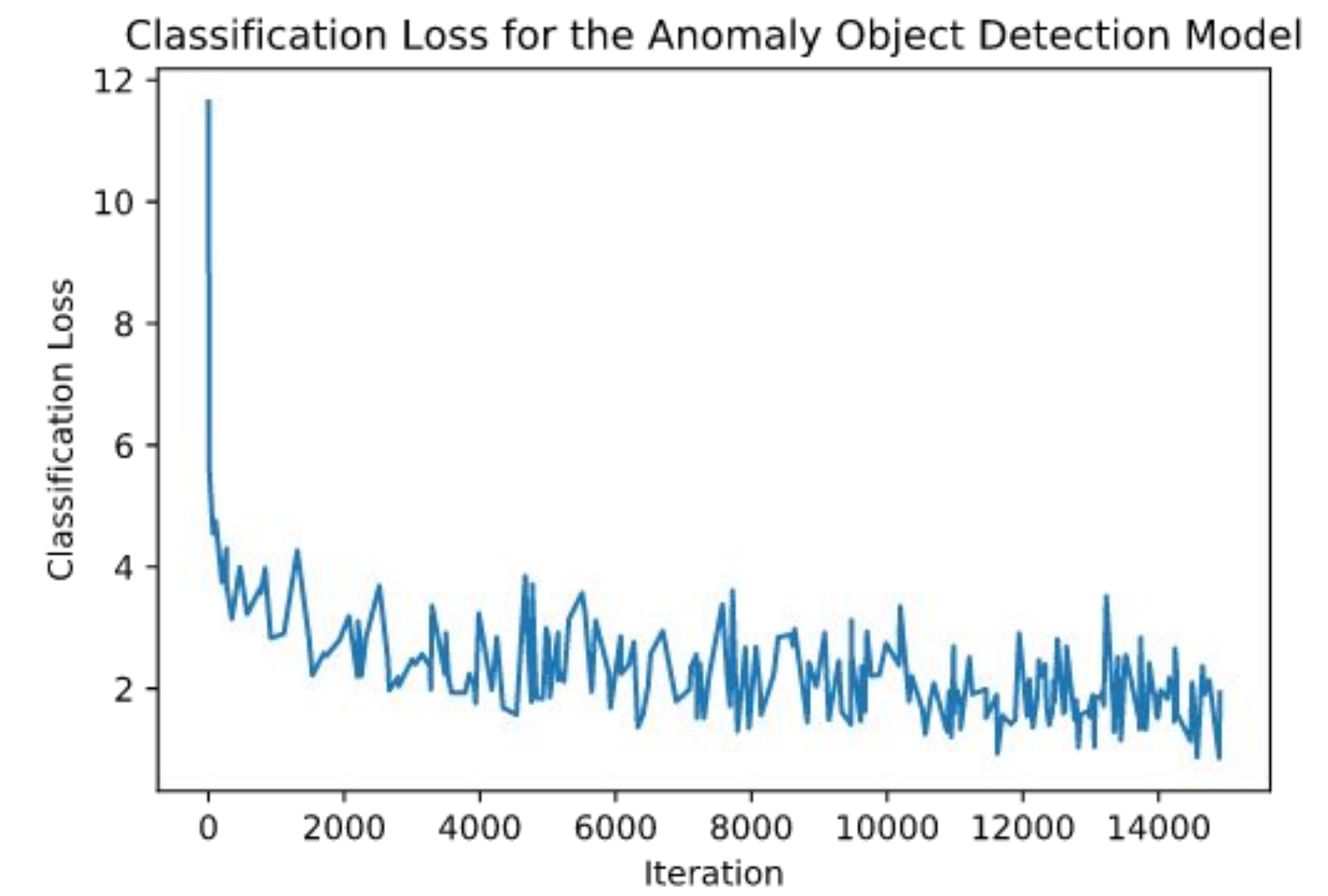
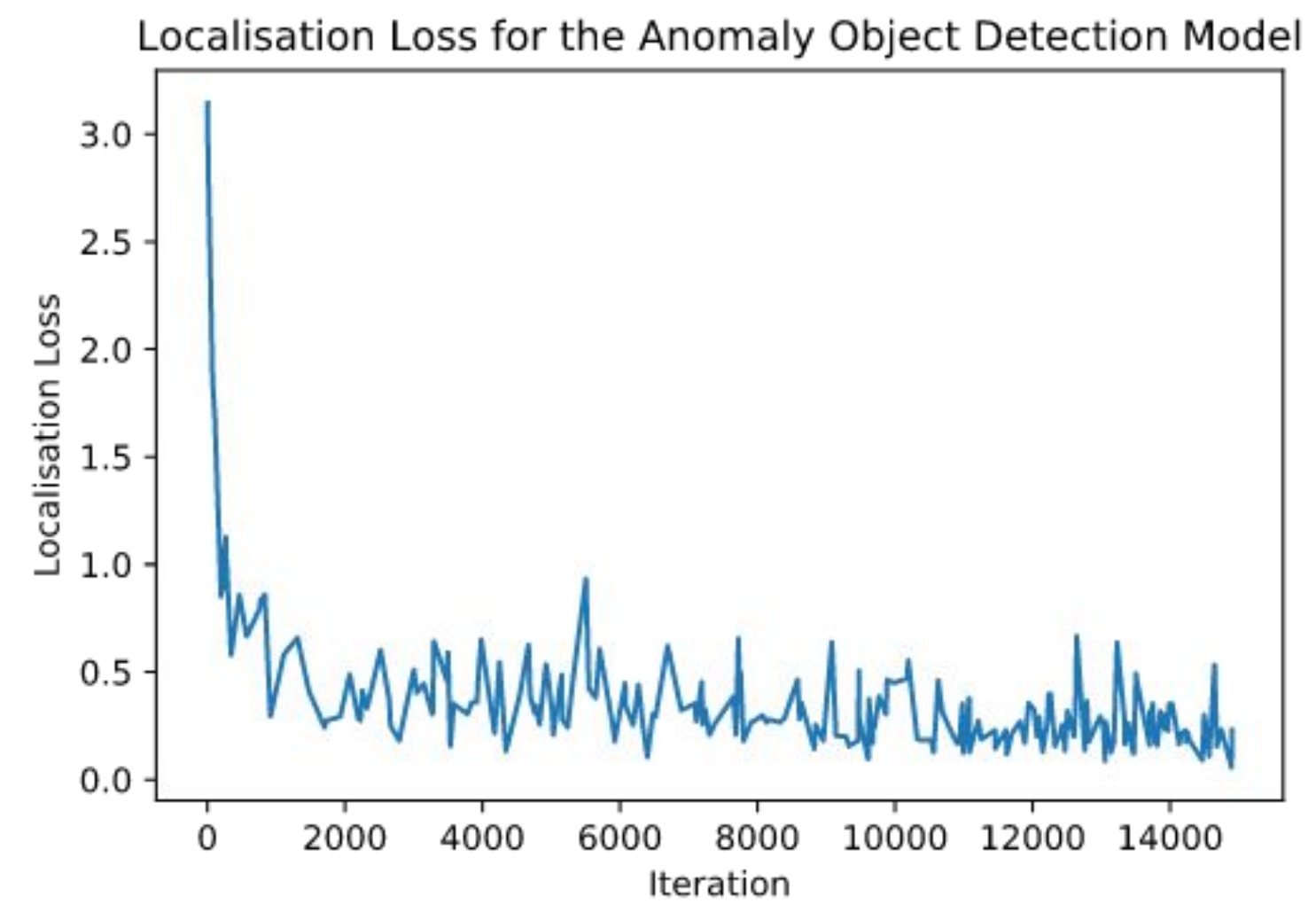
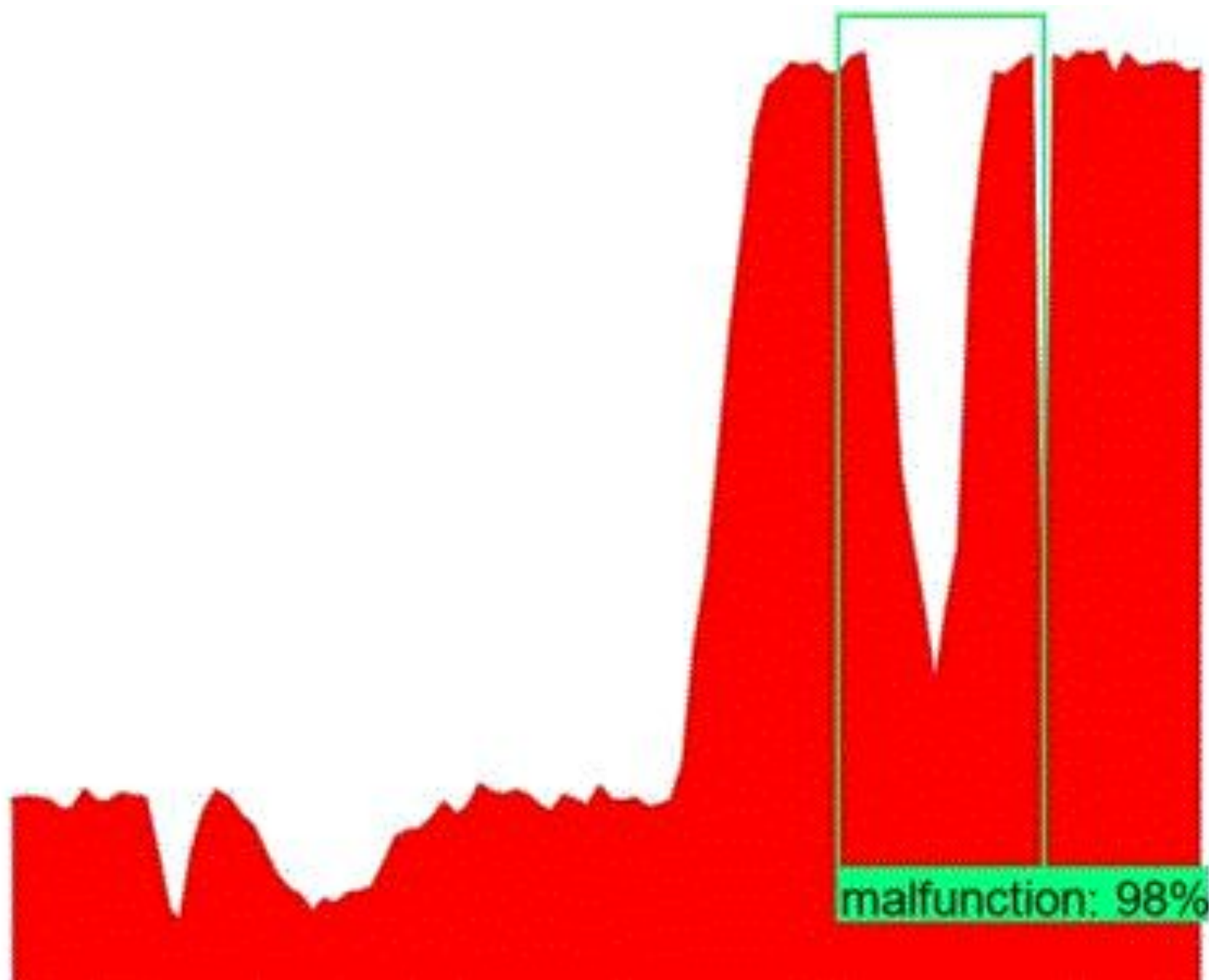
# Détection d'anomalies en tant qu'objets

## Entraînement

- Modèle pré-entraîné: MobileNet de Google (TensorFlow Object Detection API)
  - Machine utilisée: Intel i5 2.3GHz, NVIDIA GTX 960M (CUDA), 8 Go de RAM.
  - 15 000 itérations, 12 batches, 10 epochs
- 
- Avec CPU: 12 secondes/itération, arrêt de l'entraînement prévu en 264 heures (11 jours).
  - Avec GPU: 0.5 secondes/itération, arrêt de l'entraînement après ~11 heures.

# Détection d'anomalies en tant qu'objets

## Résultats







# Conclusion

- Nouveaux modèles ML
- Rudiments de la théorie du signal
- Prise en main de TensorFlow
- Techniques de génération de données



## Perspectives

- Approche combinée
- Approche combinatoire pour features de TSFresh
- Détection d'objets: Plus de données, Cluster de GPU spécialisé



# Merci

Nous tenons à remercier le Pr. Formenti pour sa patience et son aide tout au long de ce projet.