Nanomaterial-Based Sensor Array Signal Processing and TB Classification Using Machine Learning

Presented by Chenxi Liu

Supervised by Prof.Israel Cohen

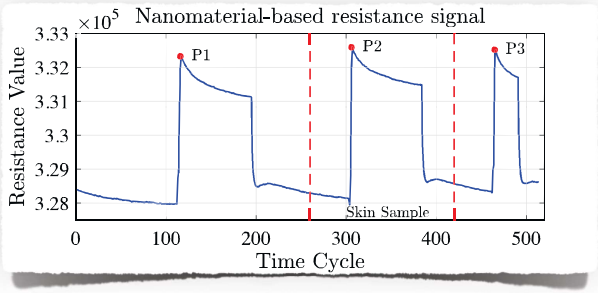
June 1st 2023

TECHNION Israel Institute of Technology

Andrew and Erna Viterbi Faculty of Electrical and Computer Engineering,

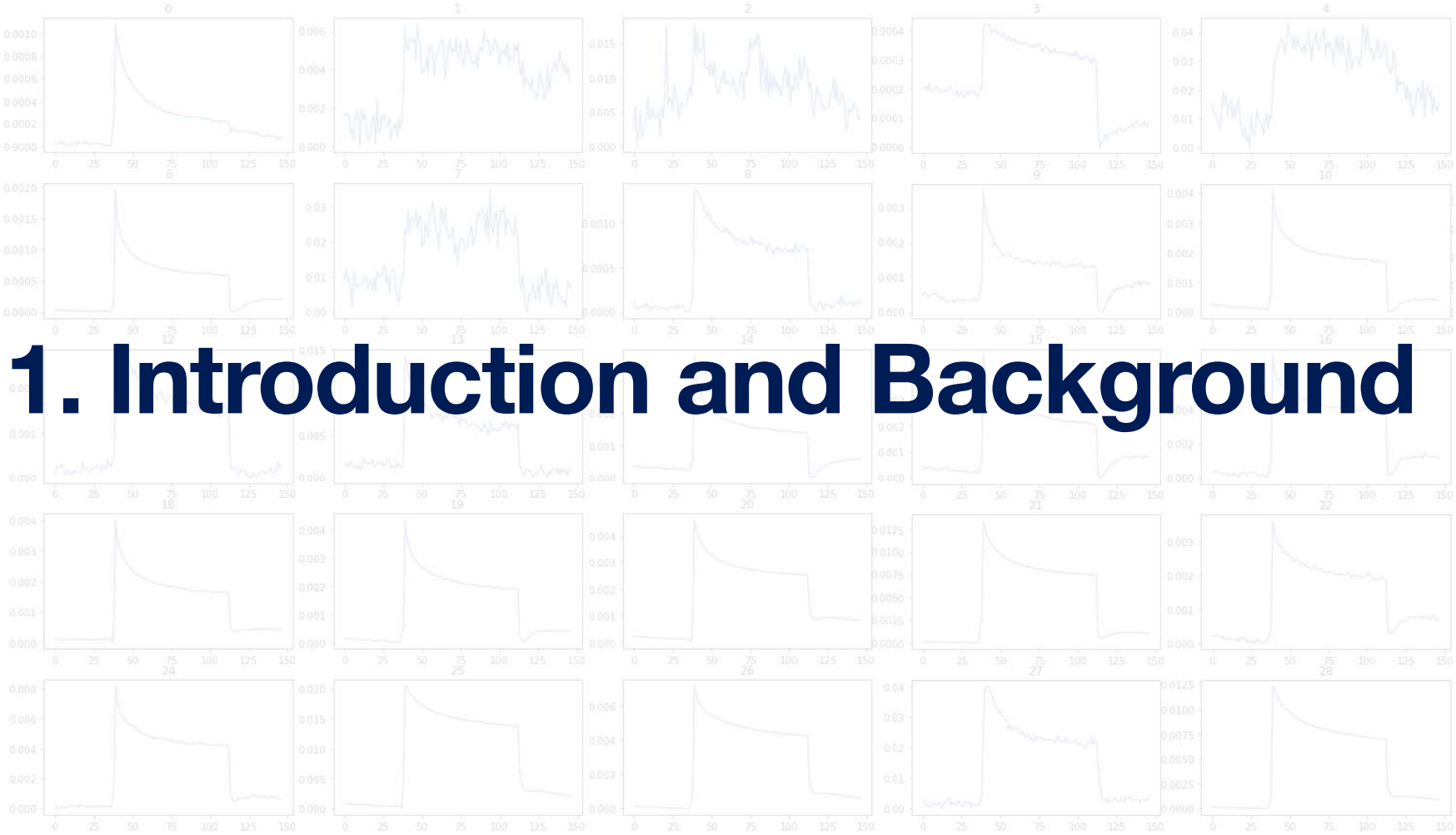
Technion — Israel Institute of Technology, Haifa, Israel.

Contents



Main Contributions

Contents



1. Introduction and Background

Introduction and Background

Tuberculosis (TB)

- An ancient, chronic disease worldwide
- Caused by: Bacillus Mycobacterium Tuberculosis
- Threatens ≈ 25% world's population
- Spread through the air. e.g. cough or sneeze

Solutions? 🤔

Yes!

🧪 Biomolecular tests

🔬 Direct microscopy test

🧬 Whole genome sequencing

🩺 Clinical signs and symptoms

Problems: 😕

- Time consumption.
- Poor clinical performance.
- Unsuitable for resource-limited settings.

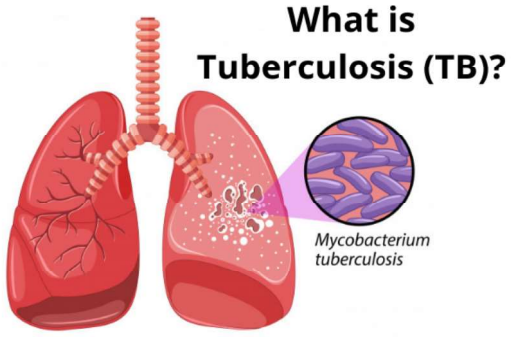
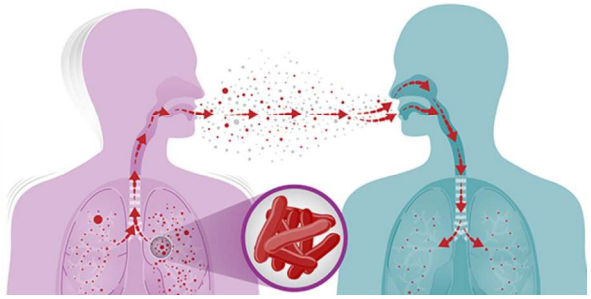


Figure 1. Healthy lung (left) and TB infected lung (right).

Infected person



Healthy person

Figure 2. TB spread from person to person through the air.

Introduction and Background

New way for TB diagnostic

- "Profiles of Volatile Biomarkers Detect Tuberculosis from Skin."
- A novel TB diagnostic pathway:
  - noninvasive, reliable, rapid

- TB-specific volatile organic compounds (VOCs)
  - Detected and quantified from the skin headspace.
  - Infected samples and healthy samples have different VOC patterns.
  - A set of nanomaterial-based sensor array will translate the collected VOCs into resistance signals.
  - Use ML models for final discrimination between ± TB.

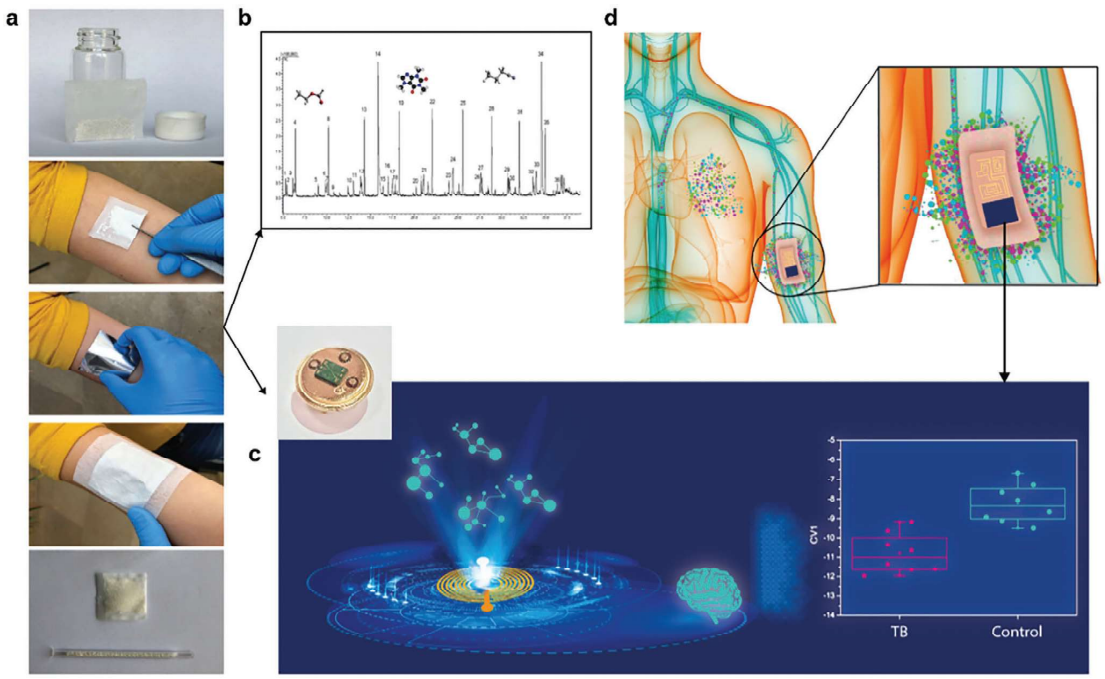


Figure 3. Study schematics.
a) Skin headspace sampling procedure with polymer.
b) GC-MS analysis of the collected samples.
c) Nanomaterial-based sensors array in conjugation machine learning analysis of the collected samples.
d) A wearable device applied directly on the skin.

"Profiles of Volatile Biomarkers Detect Tuberculosis from Skin." By Rotem Vishinkin, Hossam Haick, etal.

Introduction and Background

TECHNION Israel Institute of Technology

Small Dataset Challenges

- Success of neural networks. e.g. Computer Vision (CV), Speech, Natural Language Processing (NLP)
- 2 characteristics:
  1) Large annotated datasets
  2) Deep structure networks with large number of trainable parameters
- **Question: Can these requirements be fulfilled all the time?**

1) **Large datasets:**

- Not common in medicine, biology, chemical engineering, etc domains.
- The resource can be limited
- Obtaining and labeling data are costly
- Time consumption

2) **Deep structure networks:**

- Unsuitable for small dataset settings
- May cause overfitting problems
- Lead to poor results

Introduction and Background

Recent Studies on Solving Small Dataset Problems

- **Transfer learning:** source domain: $(\mathscr{D}_S, \mathscr{T}_S) \rightarrow$ related target domain: $(\mathscr{D}_T, \mathscr{T}_T)$.

  e.g. CV, NLP tasks.

- **Surrogate data:** generate from random numbers to imitate the original dataset distribution.

  e.g. time series analysis.

- **Our task:** Diagnosis of Tuberculosis, Machine Learning + Small Dataset Problem.

Introduction and Background

TECHNION Israel Institute of Technology

New way for TB diagnostic

- **Goal:** Develop several machine learning based networks to classify the multivariate time series sensor signals and predict their corresponding labels.

- **Key words:** Machine Learning, Small Dataset Problem, TB Classification, Multivariate Time Series Sensor Signals.

Contents

2. Data Preprocessing

Data Preprocessing

Dataset Description

- Sample Number: 928 subjects
- Labels: 0 → Active TB samples (467)
  1 → Healthy controls (461)
- Obtained Places: India, South Africa, and Latvia.
- Sensor Number: 40
- Duration: T time steps.

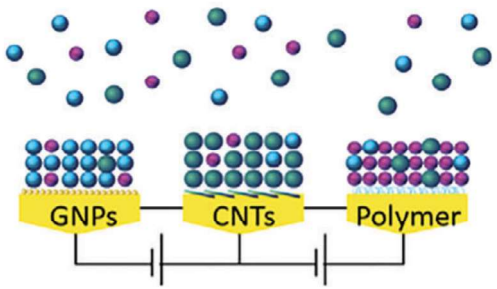


Figure 4. Schematic illustration of a sensor array.

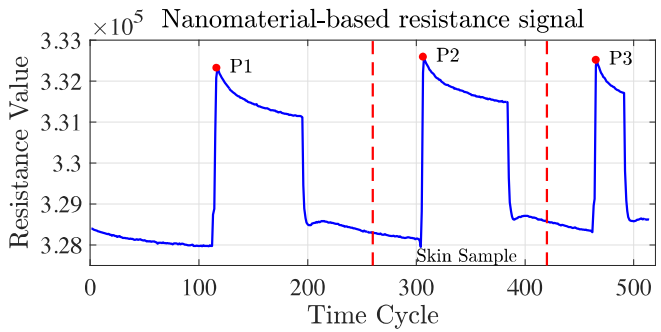


Figure 5. Full signals of one sensor, under the exposure of (1) vacuum, (2) ambient air and (3) N_2 respectively.

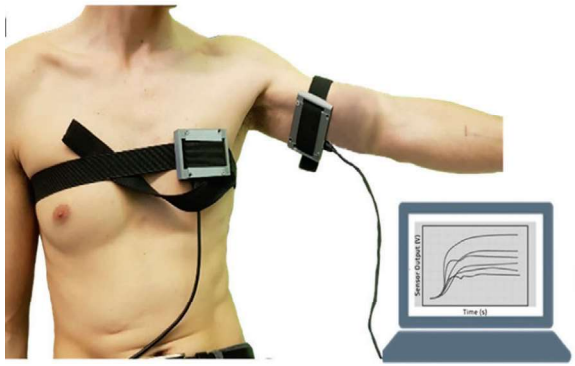


Figure 6. Wearable sensor devices that applies directly to the skin on the sample's chest and anterior arm regions.

Data Preprocessing

1. Middle Part Extraction

1. Find last 2 peak points: $P2$, $P3$.
2. Move $P2$, $P3$ to the left
3. Find the minimum length and intercept each signal into the same length.
   - The inputs have same feature dimension.

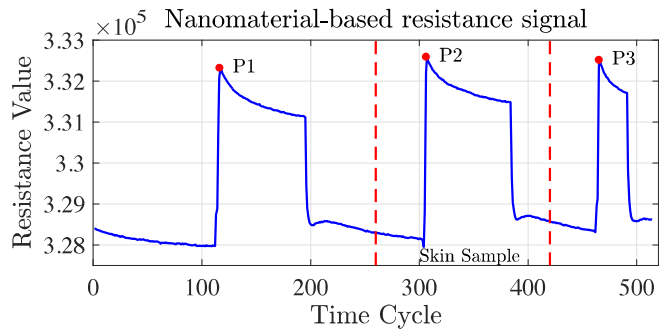


Figure 5. Full signals of one sensor, under the exposure of (1) vacuum, (2) ambient air and (3) N_2 respectively.

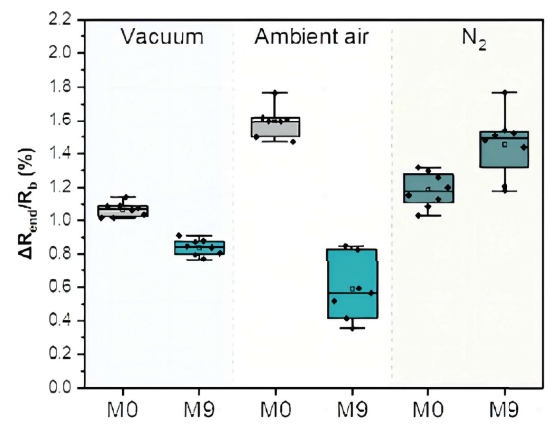


Figure 7. The change in the sensor resistivity (i.e.$\Delta R_{end}/R_b$) for different storage conditions at the starting point (M0) and after 9 months (M9).

Data Preprocessing

1. Middle Part Extraction

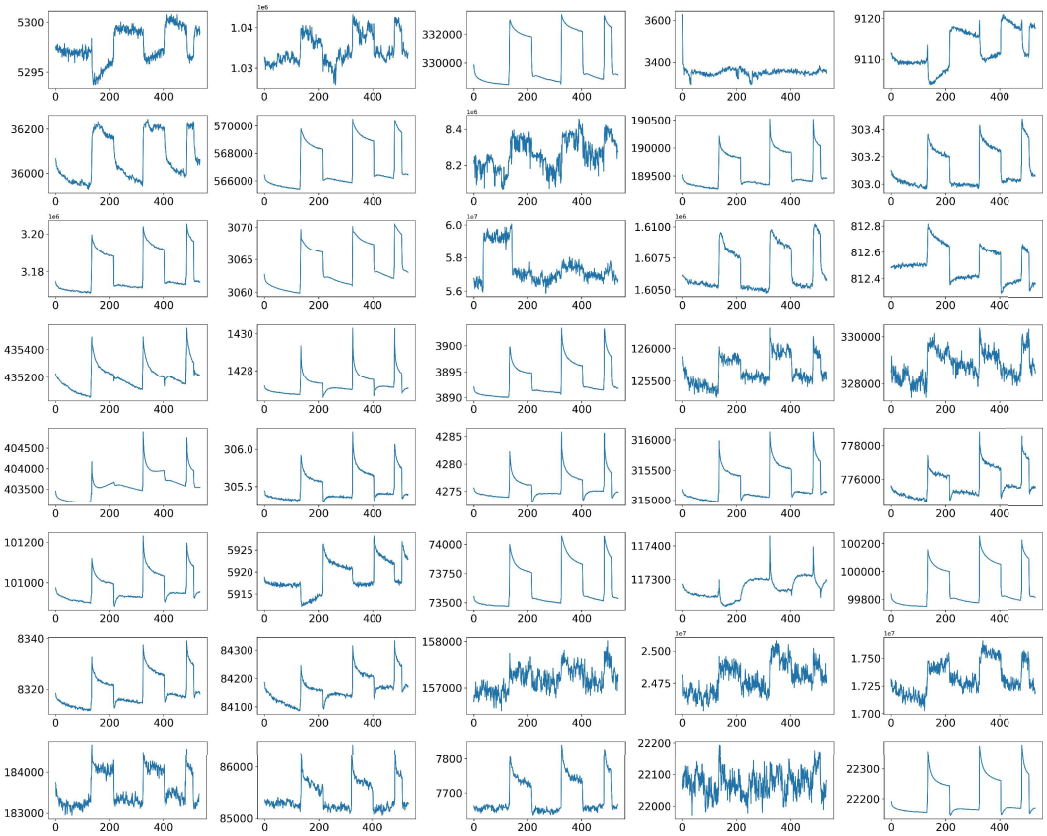


Figure 8. Original 40 sensor signals correspond to 1 sample before middle part extraction.

Middle Part Extraction

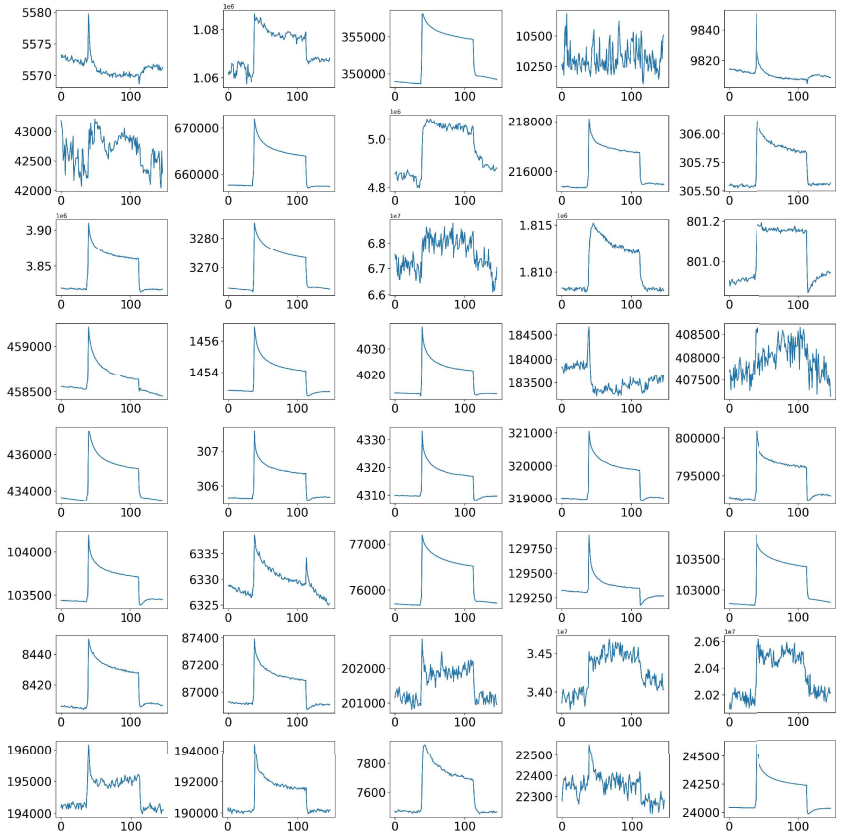


Figure 9. 40 processed sensor signals after middle part extraction.

Data Preprocessing

2. Normalization and Calibration

- **Why need normalization?**
  - Different sampling times & external environments (pressure, temperature, humidity).
  - Various starting points & resistance ranges
- **Why need calibration?**
  - Supervise the sensor's functionality
  - Overcome possible sensor response drift

- **Median Normalization:**
  - $X' = \dfrac{X}{\text{med}(X_{[0,30]})}$
  - $\hat{X} = X' - \min(X') \in [0,1]$

- **Calibration:**
  - $\bar{X} = \hat{X} \cdot \dfrac{\text{Peak}_{\hat{C}}}{\text{Peak}_{\hat{X}}}$

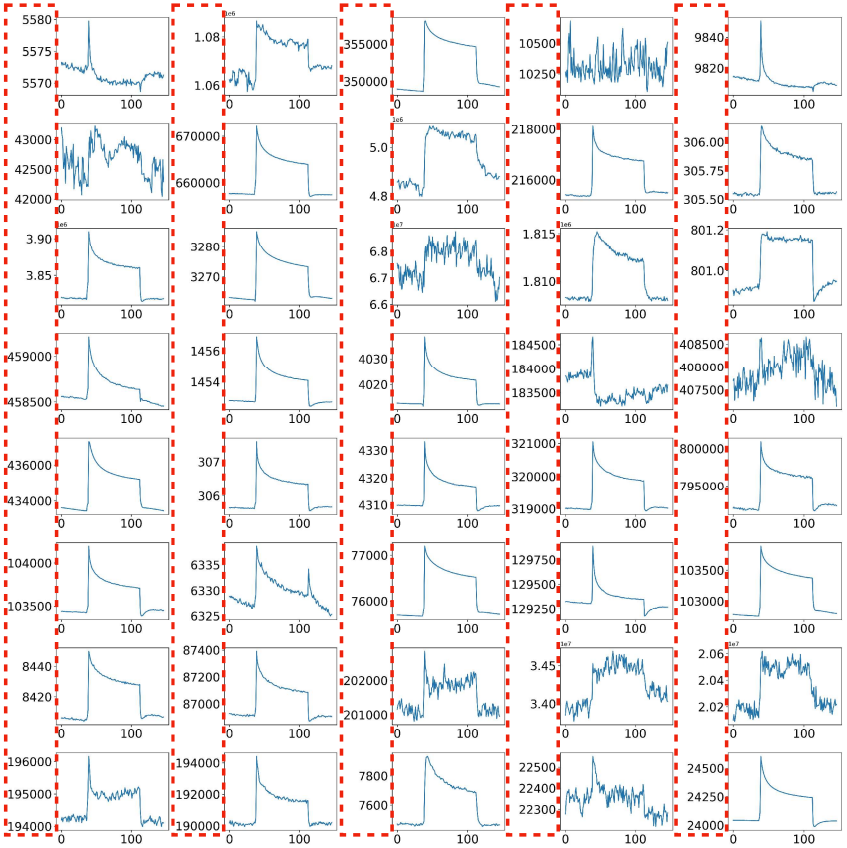


Figure 9. 40 processed sensor signals after middle part extraction.

Data Preprocessing

2. Normalization and Calibration

- **Why need normalization?**
  - Different sampling times & external environments (pressure, temperature, humidity).
  - Various starting points & resistance ranges
- **Why need calibration?**
  - Supervise the sensor's functionality
  - Overcome possible sensor response drift

- **Median Normalization:**
  - $X' = \dfrac{X}{\text{med}(X_{[0,30]})}$
  - $\hat{X} = X' - \min(X') \in [0,1]$

- **Calibration:**
  - $\bar{X} = \hat{X} \cdot \dfrac{\text{Peak}_{\hat{C}}}{\text{Peak}_{\hat{X}}}$

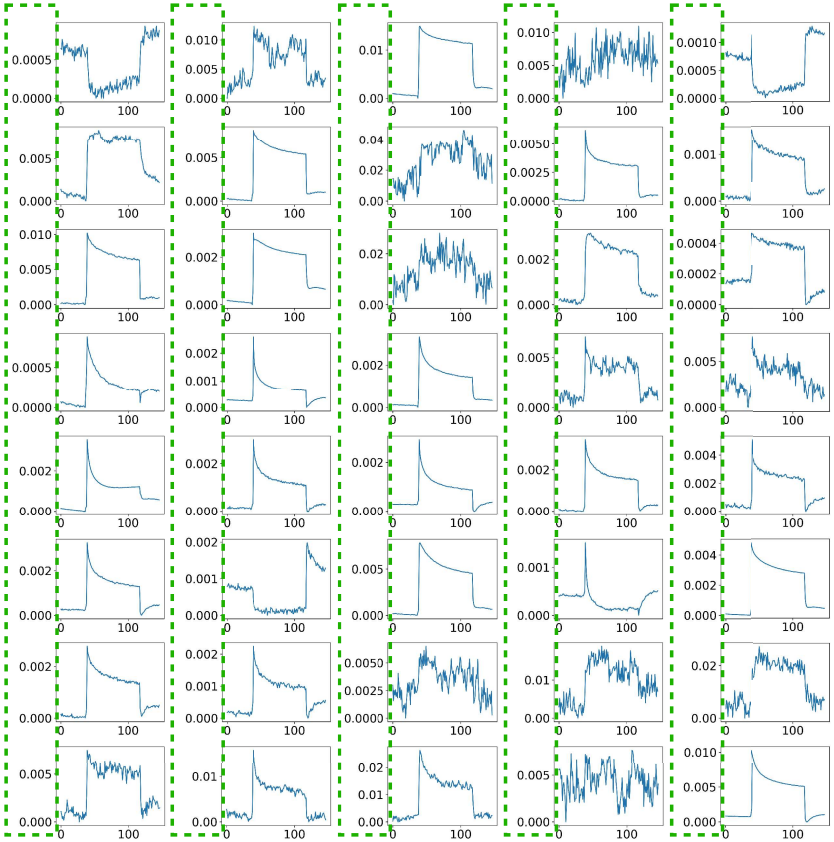


Figure 10. 40 processed sensor signals after normalization and calibration.

Data Preprocessing

3. Good sensors selection

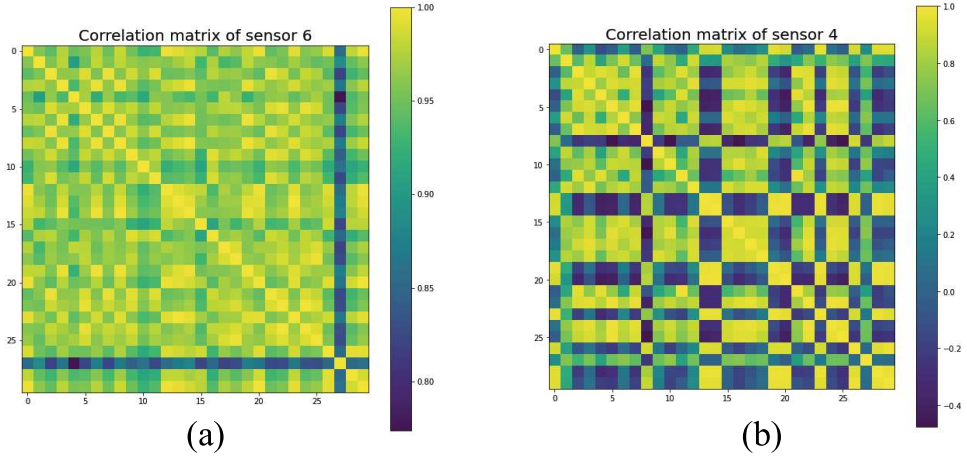


Figure 11. Pearson correlation coefficient matrix heat maps for both (a) good sensors and (b) unstable sensors.

- $$C_{xy} = \frac{\sum_{i=1}^{T}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{T}(x_i - \bar{x})^2 \sum_{i=1}^{T}(y_i - \bar{y})^2}} = \frac{\mathrm{cov}(x, y)}{\sqrt{\mathrm{var}(x) \cdot \mathrm{var}(y)}}$$
- Unstable sensors → lower similarity coefficient
- Reliable sensors → higher similarity coefficient

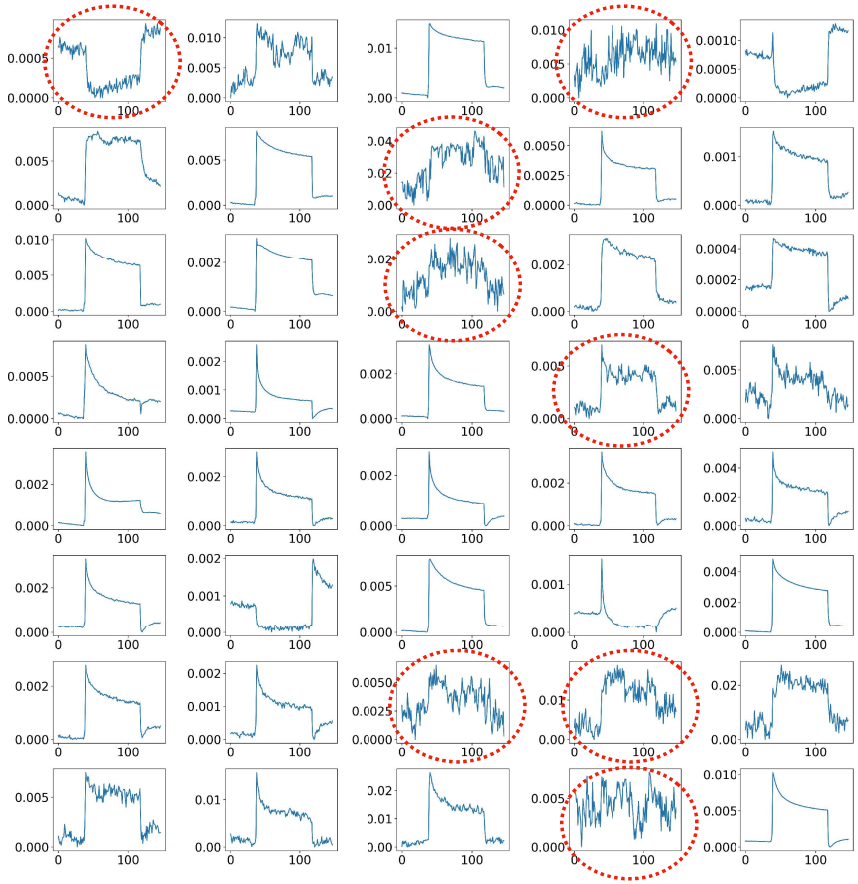


Figure 10. 40 processed sensor signals after normalization and calibration.

Data Preprocessing

3. Good sensors selection

Group A: 467 Active TB Samples, **Group B**: 461 Healthy Controls

Select 150 Random Samples / Group

Compute 150 Pearson Correlation Coefficient / Sensor

Compute Mean Coefficient Value / Sensor

Set $\theta = 0.65$

Mean Value $> \theta$ → Good Sensors (29)
Mean Value $< \theta$ → Unstable Sensors (11)

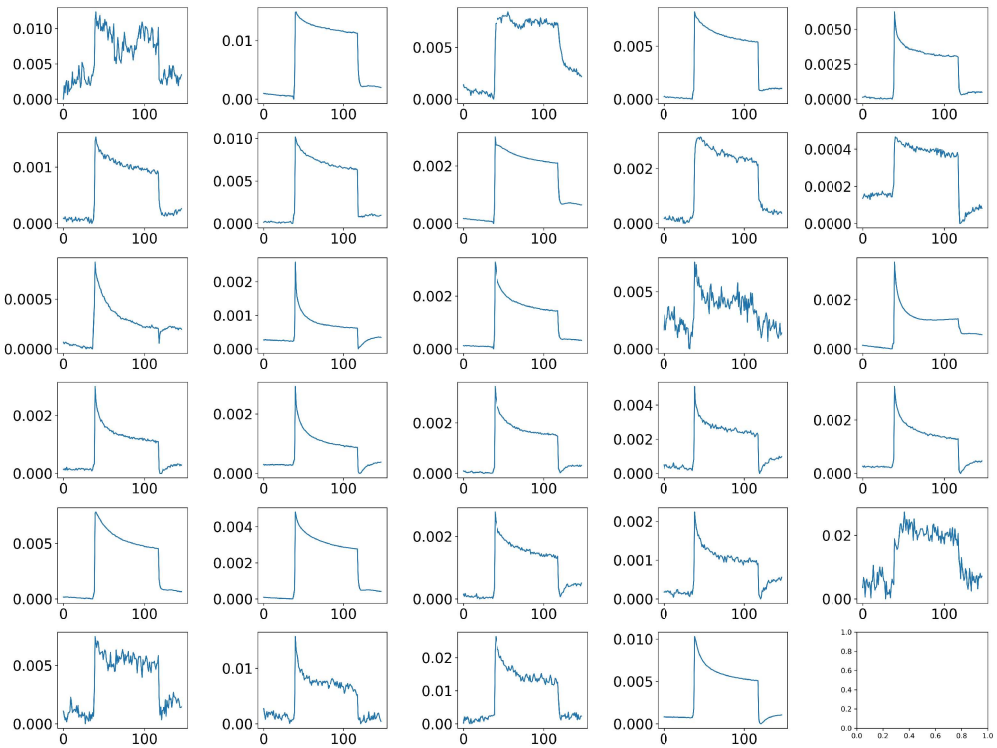
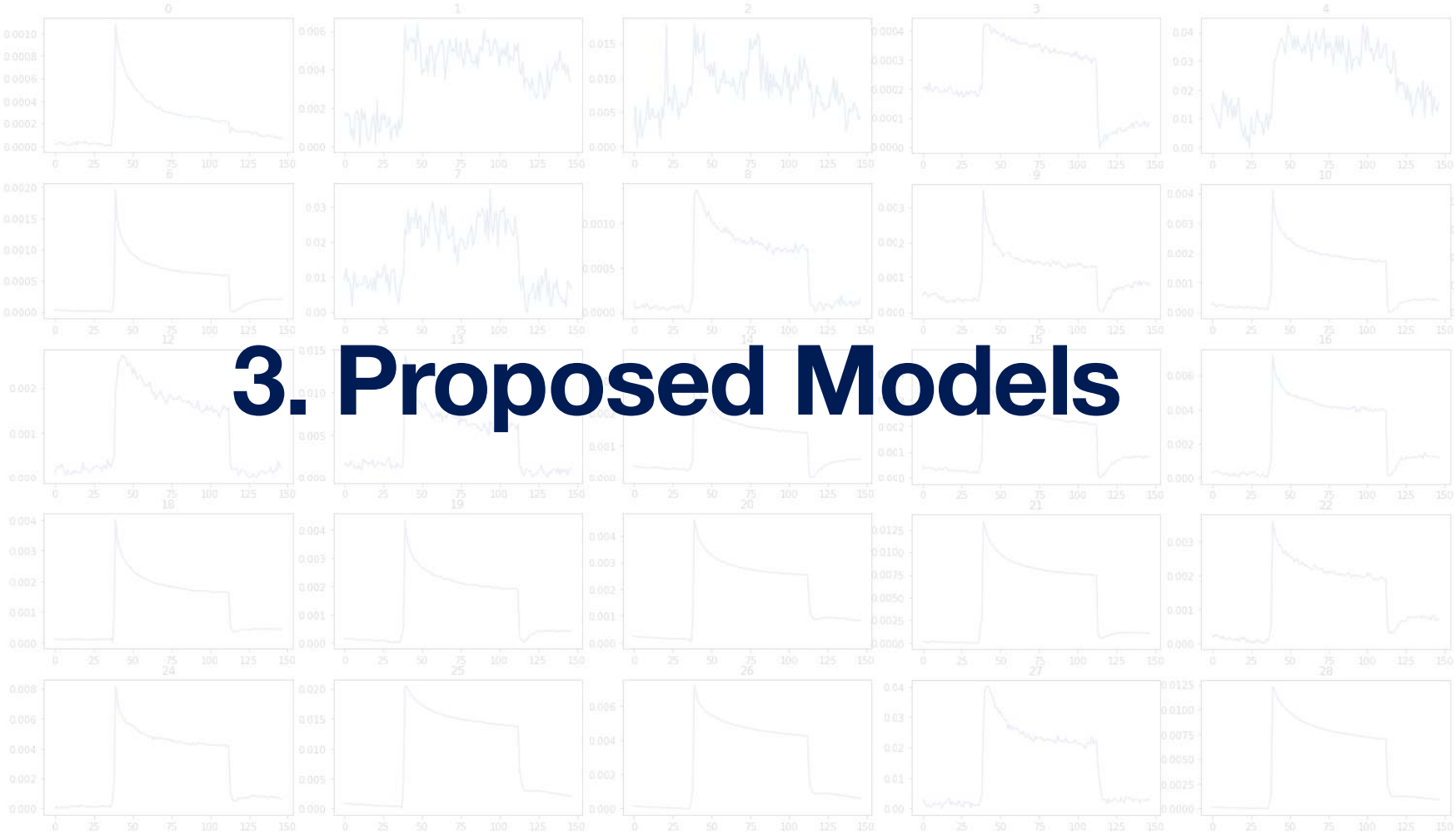


Figure 11. 29 processed sensor signals after good sensor selection.

Contents



3. Proposed Models

1. Long Short Term Memory (LSTM)

- LSTM networks are a type of recurrent neural network (RNN).
- **LSTM Network Composition:**
  - Memory vector m
  - Hidden state vector h
  - Input x
- **Benefits of using LSTM:**
  1. Reduce vanishing gradient problem
  2. Capability to handle sequences of different lengths
  3. Ability to learn from past experiences
  4. Effective at detecting complex patterns
- **Common tasks to employ LSTM:**
  - Natural Language Processing
  - Speech recognition
  - Time series analysis

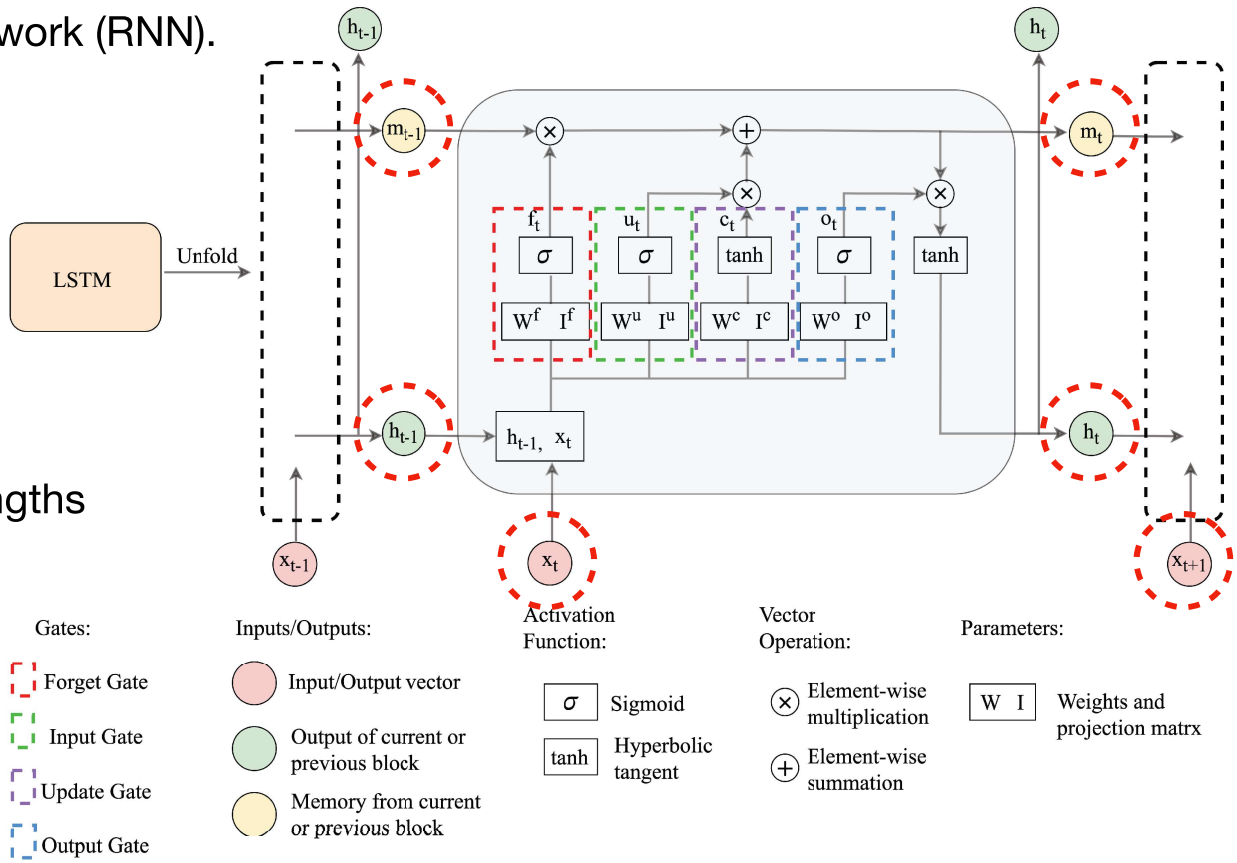
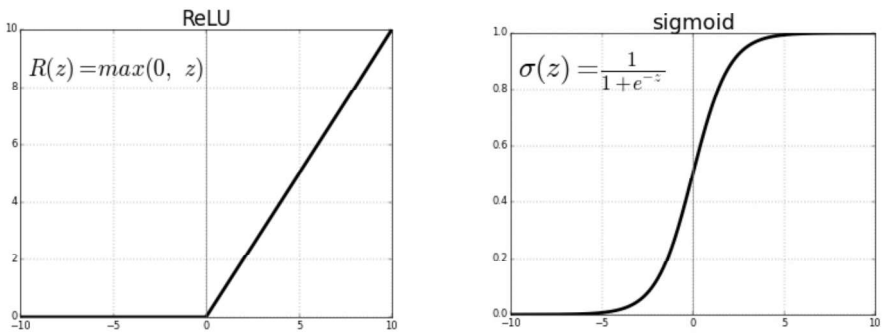


Figure 12. The structure of LSTM network.

1. Long Short Term Memory (LSTM)



(a) (b)

Figure 12. (a) ReLU and (b) Sigmoid activation function.

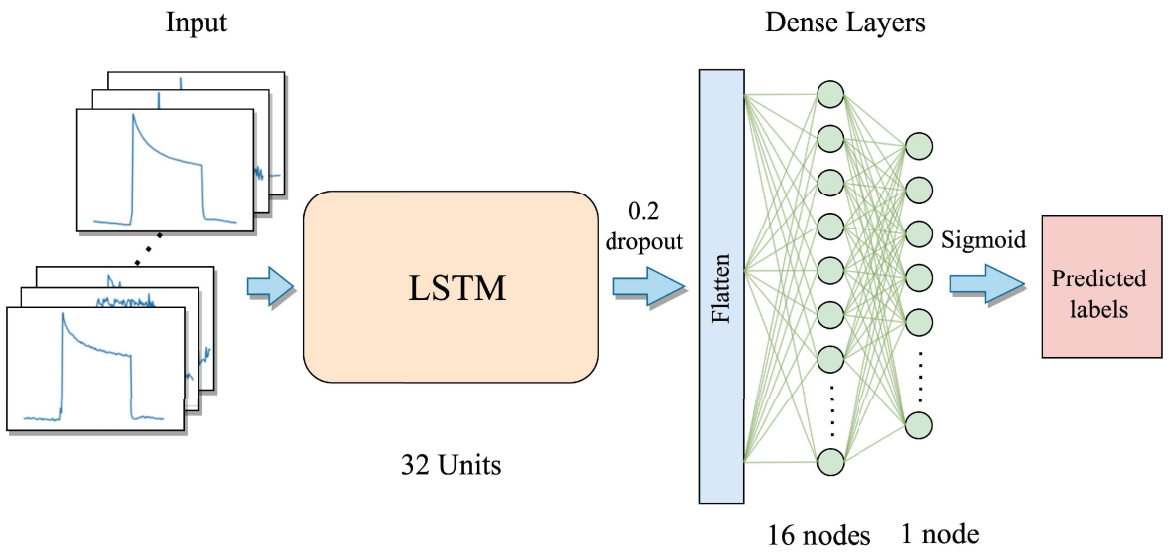


Figure 13. Proposed LSTM model architecture.

- Loss Function: Binary Cross Entropy (BCE)

$$BCE = -\frac{1}{N}\sum_{i=1}^{N}[y_i \cdot \log(p(y_i)) + (1-y_i)\cdot \log(1-p(y_i))]$$

- Input: (X, Y), $X \in \mathbb{R}^{N\times n\times T}$, $Y \in \mathbb{R}^{N}$
  - N: number of samples
  - n: number of sensors → 29
  - T: time steps → 147

2. Convolution Neural Network (CNN)

- CNN is the most widely used structure for time series classification tasks: robustness, less training time.
- **Benefits of using CNN for MTSC:**
  1. Effective in learning complex temporal patterns.
  2. Robust to variability
  3. Better performance 🆚 Support Vector Machine (SVM), Random Forrest (RF).
  4. Reduce overfitting: Pooling operation

- 1D CNN: Convolutional and Pooling layers
- Loss function: BCE loss.
- Input: (X, Y), where $X \in \mathbb{R}^{N \times n \times T}$, $Y \in \mathbb{R}^{N}$.

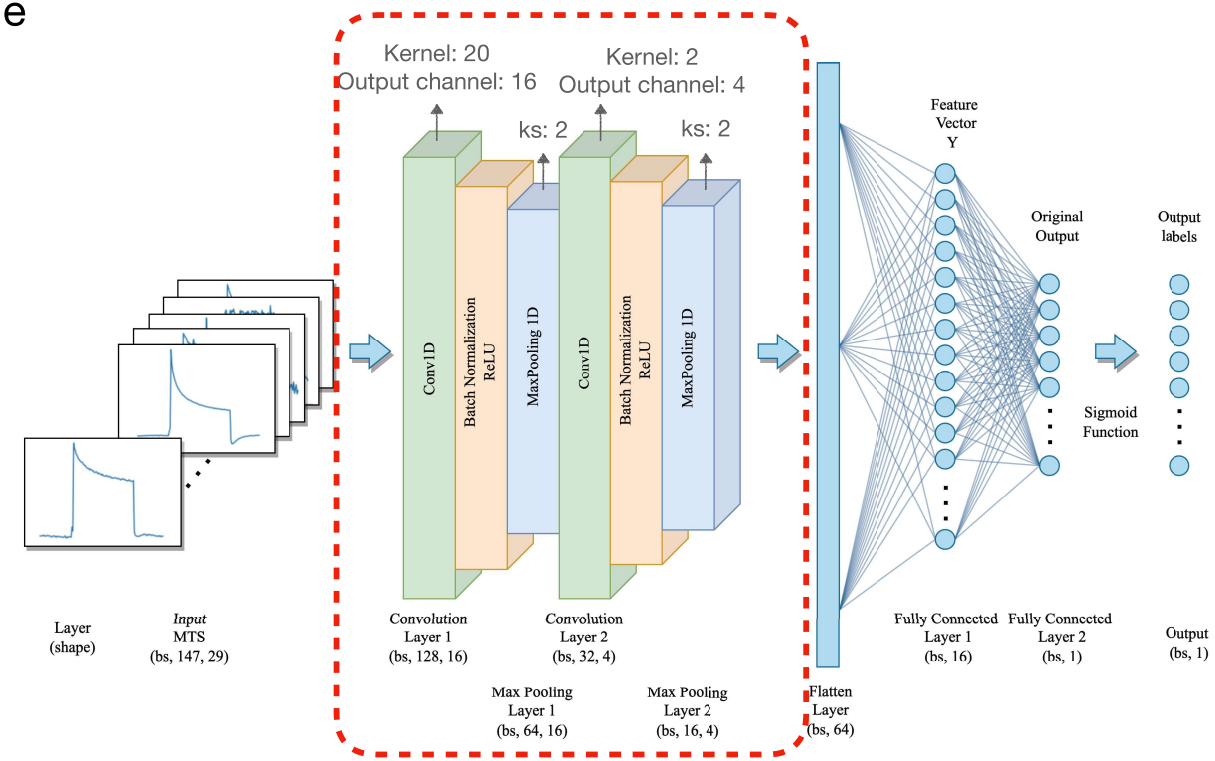


Figure 14. Proposed CNN model architecture.

3.1 Gramian Angular Field (GAF)

What is GAF?

- Encode univariate time series → 2D images.
  1. Represent time series in a polar coordinate system.
  2. Form the Gramian matrix and visualize it as 2D images.
  3. Use CNNs to identify visual patterns for prediction.

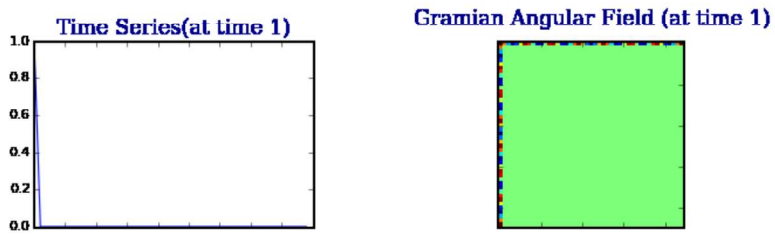


Figure 15. Known GAF values at different time steps.

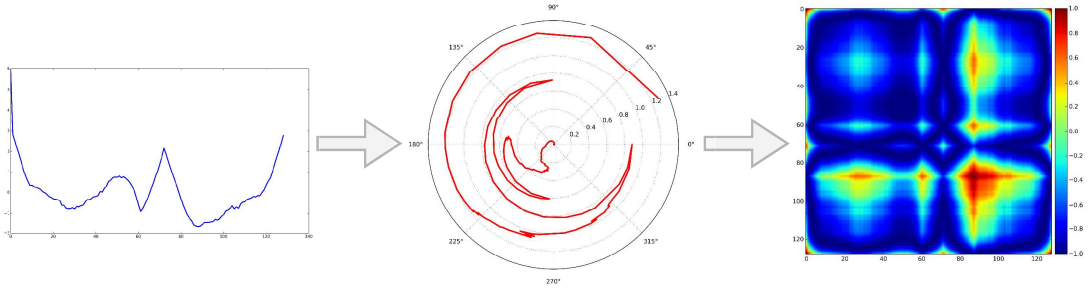
Why GAF?

- Easier to visualize and analyze the data.
- Capture cyclical patterns and correlations present in the original time series data.
- Take advantage of DL in computer vision.
- Use in a wide range of time series data.

- Financial data,
- Environmental data
- Speech data
- Electrocardiogram (ECG)
- Etc

Explanation in "reference" by Zhiguang Wang and Tim Oates.

3.1 Gramian Angular Field (GAF)



Time Series X Polar Coordinate 2D GAF Image

Figure 16. Gramian Angular Field (GAF) Transition.

- Assume we have a time series: $X = \{x_1, x_2, \cdots, x_N\}$.
  - $\hat{x}_i = \dfrac{(x_i - \max(X)) + (x_i - \min(X))}{\max(X) - \min(X)}$
  - The normalized values:

$$\hat{X} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_N\} \in [-1,1].$$

1. Convert $\hat{X}$ into polar coordinates:
   - $\phi_i = \arccos \hat{x}_i, \ -1 \le \hat{x}_i \le 1, \ \hat{x}_i \in \hat{X}$
   - $r_i = \dfrac{t_i}{N}, \ t_i \in \mathbb{N}$
2. The GAF method defines its own "special" inner product as:
   - $\langle \hat{x}_1, \hat{x}_2 \rangle = \cos(\phi_1 + \phi_2)$

$$= \hat{x}_1 \cdot \hat{x}_2 - \sqrt{1 - \hat{x}_1^2} \cdot \sqrt{1 - \hat{x}_2^2}$$

- $$G = \begin{pmatrix} \cos(\phi_1 + \phi_1) & \cos(\phi_1 + \phi_2) & \cdots & \cos(\phi_1 + \phi_n) \\ \cos(\phi_2 + \phi_1) & \cos(\phi_2 + \phi_2) & \cdots & \cos(\phi_2 + \phi_n) \\ \vdots & \vdots & \ddots & \vdots \\ \cos(\phi_n + \phi_1) & \cos(\phi_n + \phi_2) & \cdots & \cos(\phi_n + \phi_n) \end{pmatrix}$$

$$= \hat{X}' \cdot \hat{X} - \sqrt{I - \hat{X}^2}' \cdot \sqrt{I - \hat{X}^2}$$

Image and explanation in "reference" by Zhiguang Wang and Tim Oates.

3.2 Gramian Angular Field-CNN (GAF-CNN)

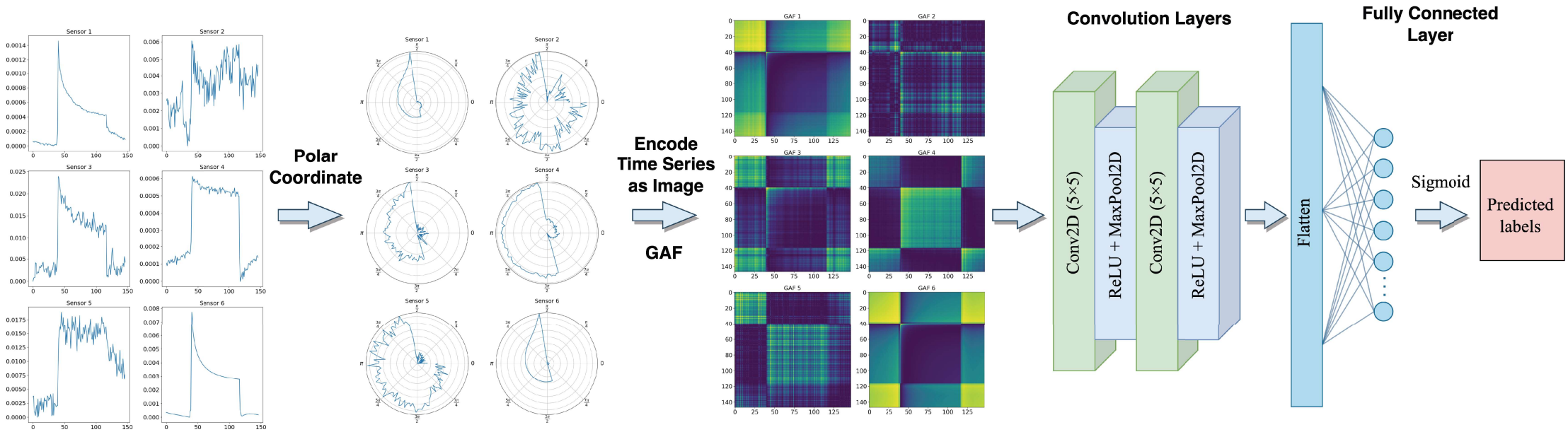
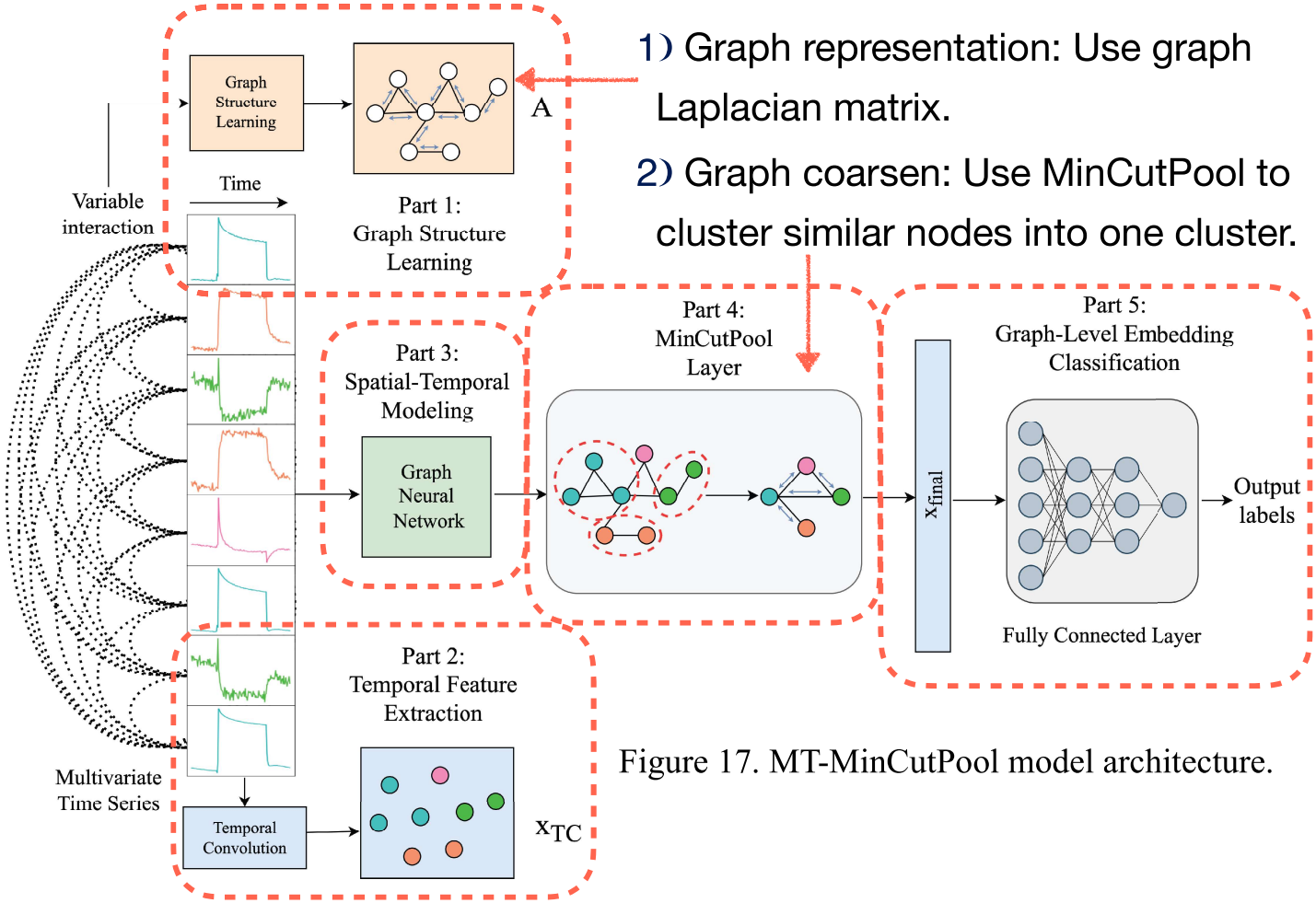


Figure 16. GAF-CNN model architecture.

- $X \in \mathbb{R}^{N \times n \times T} \rightarrow X_{GAF} \in \mathbb{R}^{N \times T \times T \times n}$: (batch size, height, width, channels).
- Difference: 2D convolutional layers and 2D max pooling layers.

4. MT-MinCutPool



1) Graph representation: Use graph Laplacian matrix.

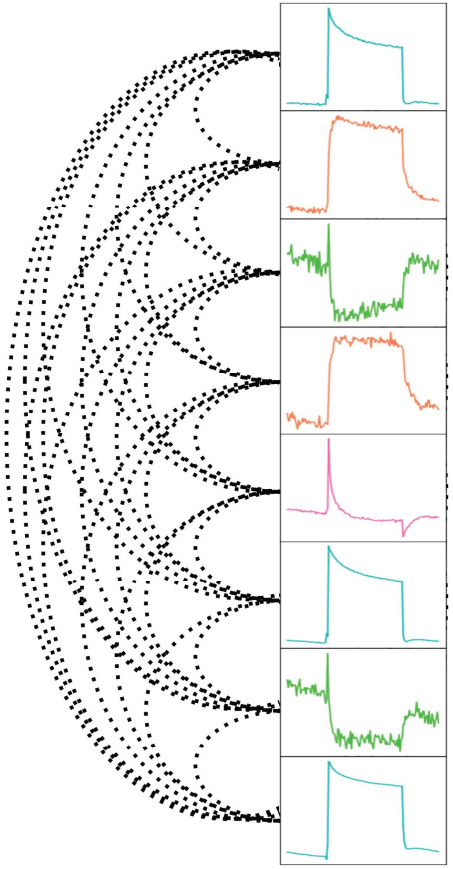
2) Graph coarsen: Use MinCutPool to cluster similar nodes into one cluster.

Figure 17. MT-MinCutPool model architecture.

- In paper: the authors proposed a novel graph pooling based framework **"Multivariate Time Series Classification with Variational Graph Pooling (MTPool)".**
  - Obtain an expressive global representation of MTS.
  - Adopt graph neural network (GNN) to solve MTSC problem.
- We propose MT-MinCutPool model, a modified MTPool.
- 2 Main differences.

4. MT-MinCutPool

Part 1. Graph Structure Learning



- Euclidean distance: prone to sensitivity and distortion in time axis.
- Dynamic Time Warping (DTW) distance:
  - Effective distance measurement for time series problems.
  - Non-linear alignment of similar time series that are locally out of phase.
- For subsequent computing:

  1) Construct distance matrix $DTW \in \mathbb{R}^{N \times n \times n}$

  2) Set a threshold θ

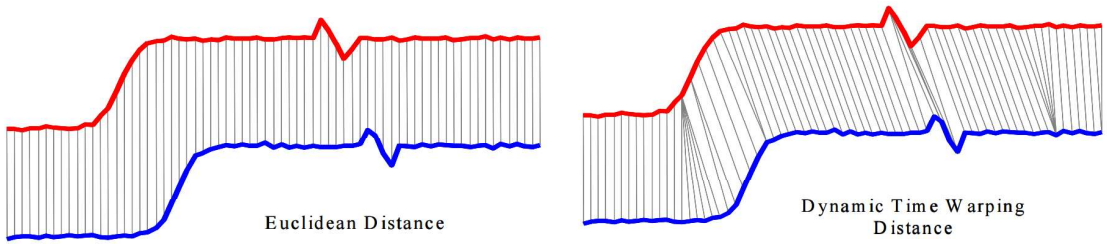
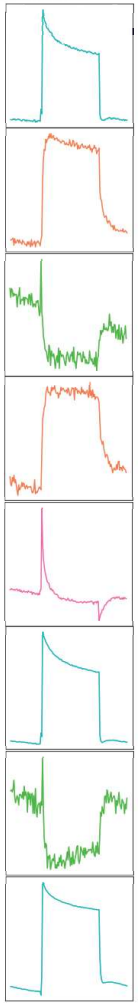
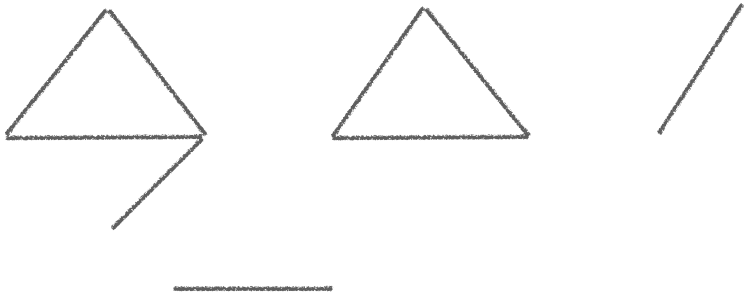


Figure 18. An illustration between Euclidean matching and DTW matching from paper.

4. MT-MinCutPool

Graph Structure Learning

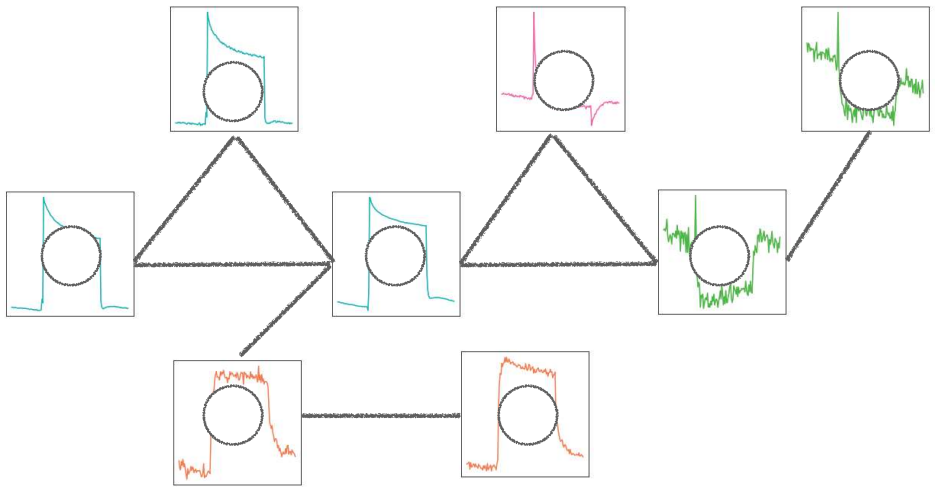




1. Build Adjacency Matrix A:
   - $A_{ij} = \begin{cases} 1, & \text{if DTW[i, j]} < \theta\,, \\ 0, & \text{if DTW[i, j]} > \theta\,. \end{cases}$
2. Build Degree Matrix D:
   - the row sum of A
3. Build Laplacian Matrix L:
   - $L = D - A$

4. MT-MinCutPool

Part 1. Graph Structure Learning



1. Build Adjacency Matrix A:
   - $A_{ij} = \begin{cases} 1, & \text{if DTW[i, j]} < \theta\,, \\ 0, & \text{if DTW[i, j]} > \theta\,. \end{cases}$
2. Build Degree Matrix D:
   - the row sum of A
3. Build Laplacian Matrix L:
   - $L = D - A$

4. MT-MinCutPool

Part 1. Graph Structure Learning

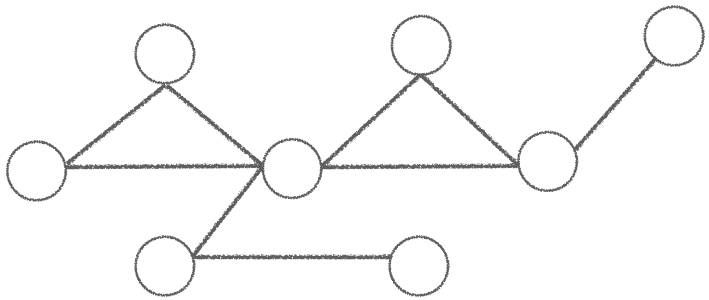


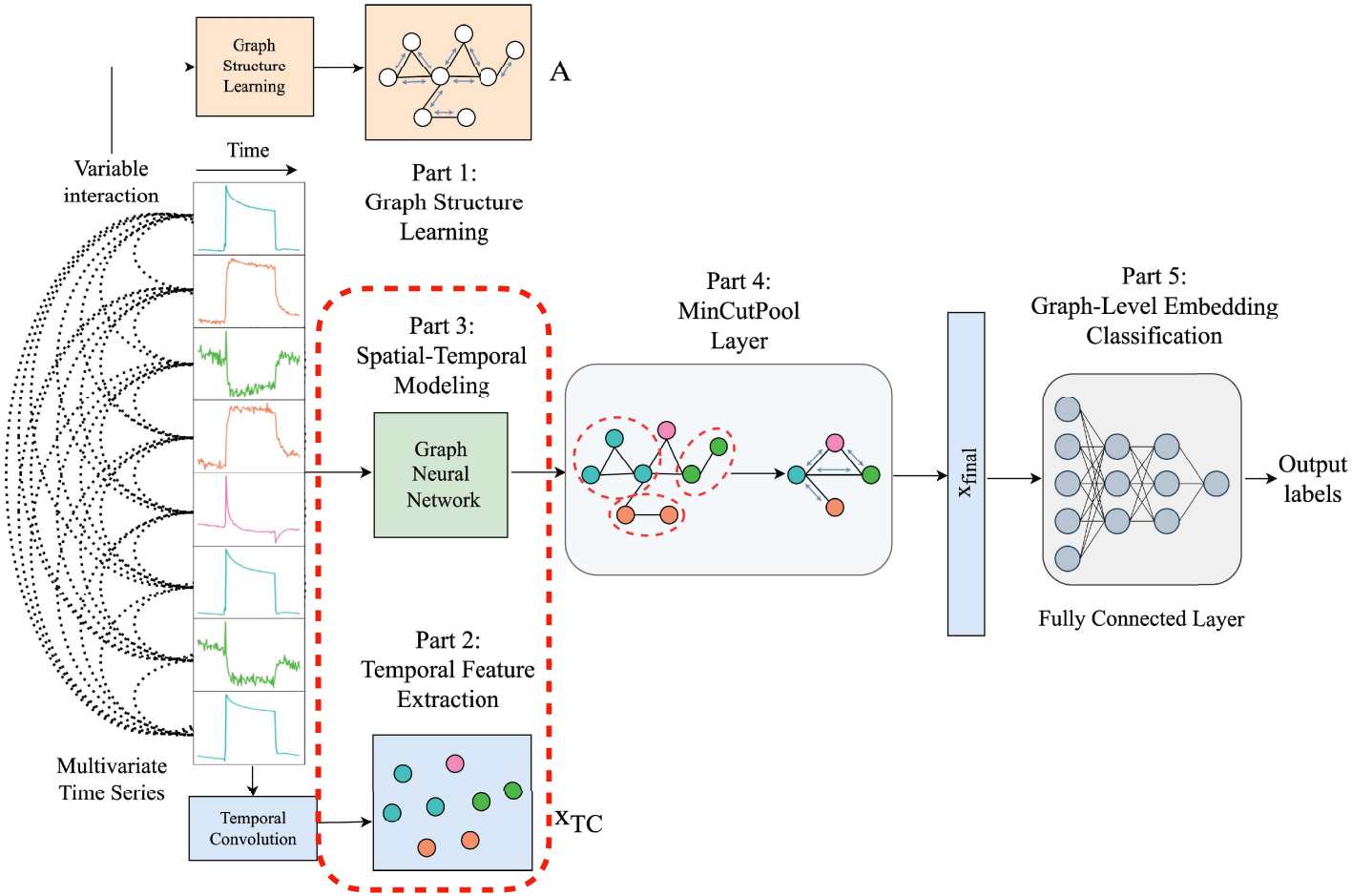
Figure 18. MT-MinCutPool model graph structure.

Algorithm 3.1 Build Laplacian Adjacency matrix.

Input: $\mathrm{X} \in \mathbb{R}^{N \times n \times T}$, θ
Output: $\mathrm{L} \in \mathbb{R}^{N \times n \times n}$

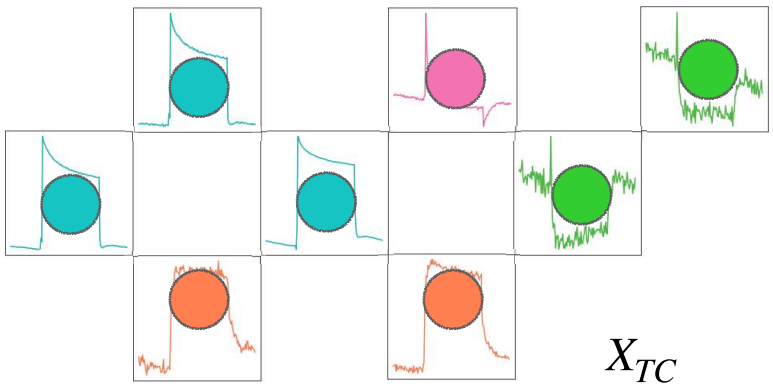
(1) Build Distance Matrix DTW
DTW $\leftarrow$ empty matrix with shape $\mathbb{R}^{N \times n \times T}$
for $i = 1$ to N **do**
  x $\leftarrow$ X[i]
  distance $\leftarrow$ empty matrix with shape $\mathbb{R}^{n \times n}$
  for $j = 1$ to n **do**
    for $k = 1$ to n **do** distance[j, k] $\leftarrow$ dtw(x[j], x[k]) {dtw($\cdot$) is the dynamic time warping distance function}
    end for
  end forDTW[i] $\leftarrow$ distance
end for
(2) Build Degree Matrix D
D $\leftarrow$ empty array with shape $\mathbb{R}^{N \times n \times T}$
A $\leftarrow$ int(DTW $< \theta$)
for $i = 1$ to N **do**
  adj $\leftarrow$ A[i]
  D[i] $\leftarrow$ diagonal($\sum_{k=1}^{n}$ adj[k, :]) {diagonal($\cdot$) is the diagonalized function}
end for
(3) Build Laplacian Matrix L
L = D $-$ A

4. MT-MinCutPool

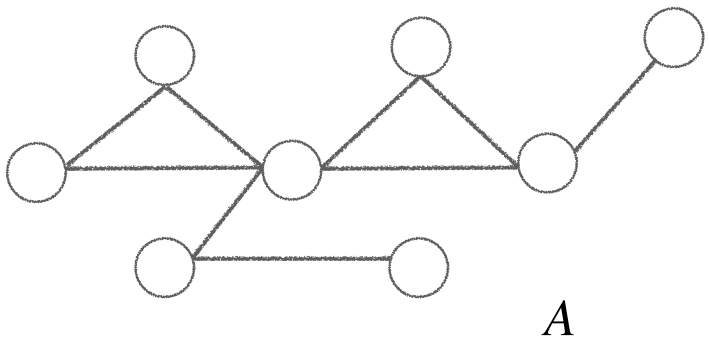


4. MT-MinCutPool

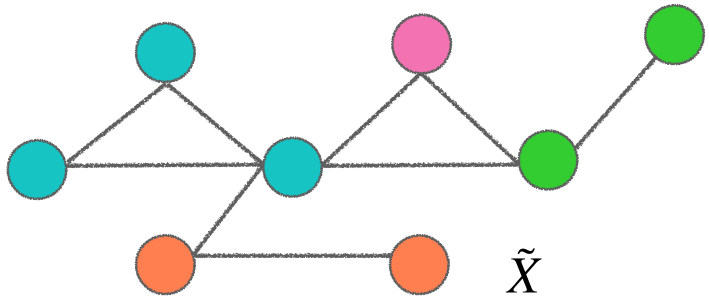
Part 2. Temporal Feature Extraction + Part 3. Spatial-Temporal modeling



+



=



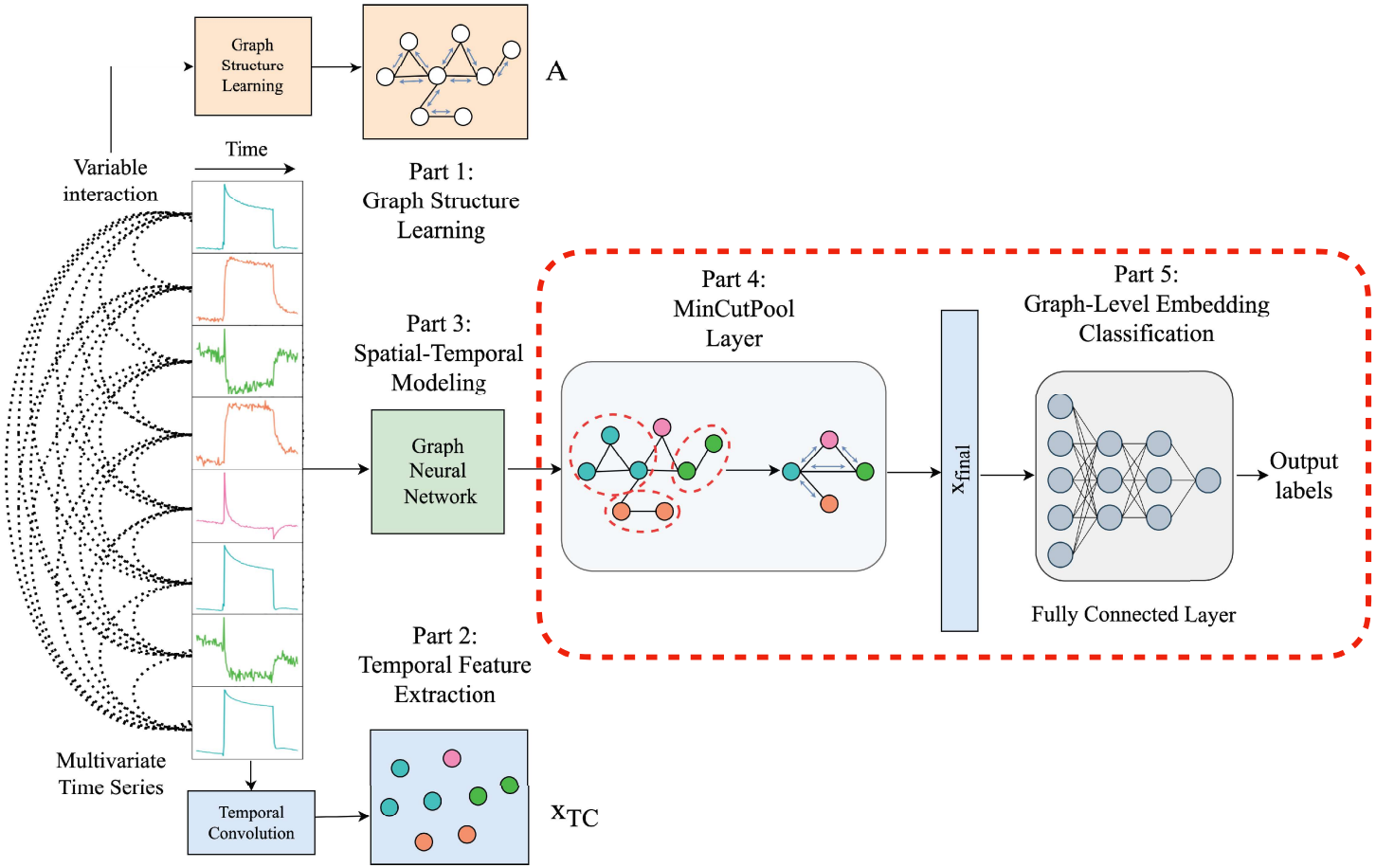
Part 2: $X_{TC} = ||_{i=1}^{m} f_i = ||_{i=1}^{m} \sigma(W_i * X + b)$

- $||_{i=1}^{m}$ concatenation operation.
- f_i output feature maps of each convolution layer.

Part 3: $\tilde{X} = G(A, X_{TC}, W, b) = \sigma(A * X_{TC} * W + b)$

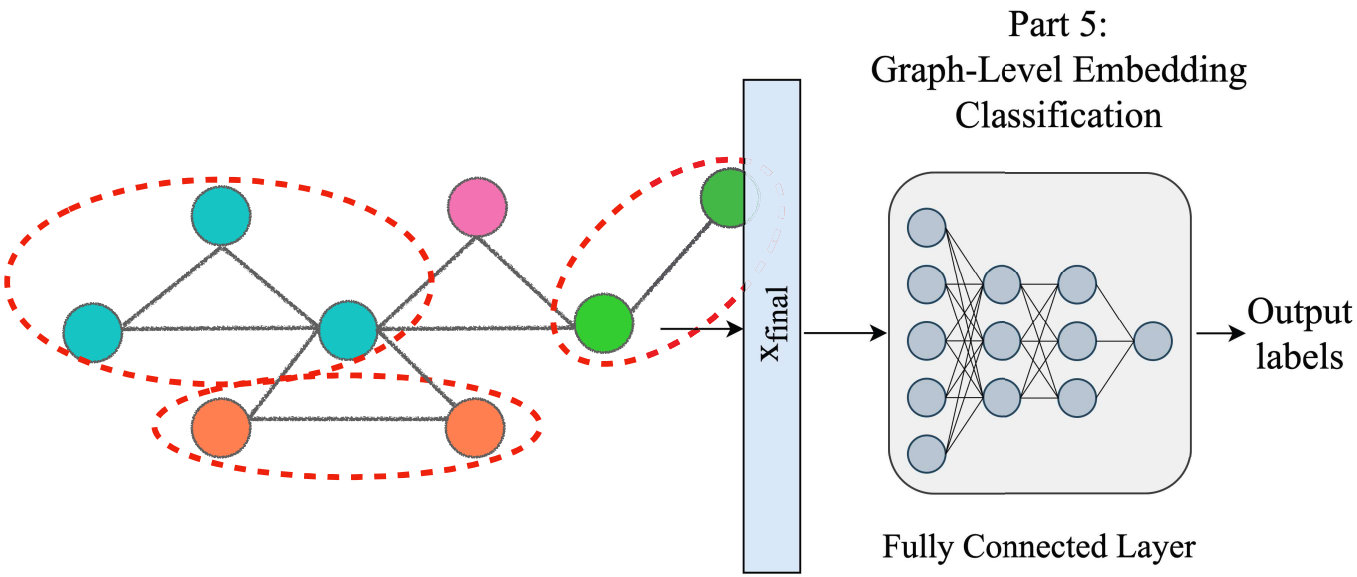
- $G(\cdot)$ graph convolution function.

4. MT-MinCutPool



4. MT-MinCutPool

Part 4. MinCutPool Layer + Part 5. Graph Level Embedding Classification



Part 4: Graph coarsen

1) Cluster assignment matrix S

- $S = MLP(\bar{X}, W_{MLP})$

2) Pooling by MinCutPool:

- Normalized adjacency matrix:

$$\tilde{A} = D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \in \mathbb{R}^{N \times n \times n}$$

- Coarsened adjacency matrix:

$$A_{pool} = S^T \tilde{A} S$$

- Coarsened feature matrix:

$$X_{pool} = S^T X$$

Model Summary

Layer	Stride	Activation	Kernel Size	Input Shape	Output Shape	Parameter Number
Conv1D	1	ReLU	20	(bs, 147, 29)	(bs, 128, 16)	9296
MaxPooling1D	2	-	2	(bs, 128, 16)	(bs, 64, 16)	0
Conv1D	2	ReLU	2	(bs, 64, 16)	(bs, 32, 4)	2116
MaxPooling1D	2	-	2	(bs, 32, 4)	(bs, 16, 4)	0
Flatten	-	-	-	(bs, 16, 4)	(bs, 64)	0
Dense	-	ReLU	-	(bs, 64)	(bs, 16)	1040
Dense	-	Sigmoid	-	(bs, 16)	(bs, 1)	17

Table 2. CNN model layers and parameters.

Layer	Output Shape	Activation	Parameter Number
LSTM	(Batch size, 32)	-	7936
Dropout	(Batch size, 32)	-	0
Dense	(Batch size, 16)	ReLU	528
Dense	(Batch size, 1)	Sigmoid	17

Table 1. LSTM model layers and parameters.

Layer	Stride	Activation	Kernel Size	Input Shape	Output Shape	Parameter Number
Conv2D	1	ReLU	(5, 5)	(bs, 147, 147 29)	(bs, 143, 143, 12)	8712
MaxPooling2D	2	-	(2, 2)	(bs, 143, 143, 12)	(bs, 71, 71, 12)	0
Conv2D	1	ReLU	(5, 5)	(bs, 71, 71, 12)	(bs, 67, 67, 6)	1806
MaxPooling2D	2	-	(2, 2)	(bs, 67, 67, 6)	(bs, 33, 33, 6)	0
Flatten	-	-	-	(bs, 33, 33, 6)	(bs, 6534)	0
Dense	-	Sigmoid	-	(bs, 6534)	(bs, 1)	6535

Table 3. GAF-CNN model layers and parameters.

Model	Parameter Number
LSTM	8481
CNN	12469
GAF-CNN	17053

Table 4. Total number of parameters of the first 3 proposed models.

Model Summary

Common Characteristics:

1. Limited number of parameters.
2. Lightweight models with limited layers.
3. Employed regularization techniques:
   - LSTM: Dropout layer
   - CNN, GAF-CNN: Max pooling layer
   - MT-MinCutPool: MinCutPool pooling layer

Technique Benefits:

- Improve generalization.
- Make the model to learn more robust features.
- Reduce spatial size of the feature map.
- Reduce the number of model parameters.
- Avoid overfitting problems.

Contents

TECHNION
Israel Institute
of Technology

4. Results and Discussion

Experiment Setup

TECHNION Israel Institute of Technology

Model Training

- TB samples:
  - 80% for training and validation
  - 20% for testing
- 5 fold stratified cross-validation
- 10 runs for each model
- Results:
  1) Choose the best model with the highest validation accuracy
  2) Evaluate on the test set / run
  3) Compute mean test accuracy from 10 runs as the final result.

Evaluation Metrics

- $Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$
- $Sensitivity = \frac{TP}{TP + FN}$
- $Specificity = \frac{TN}{TN + FP}$
- Area Under the ROC Curve (AUC)

- TP: True Positive
- FP: False Positive
- TN: True Negative
- FN: False Negative

Results and Discussion

		LSTM	CNN	GAF-CNN	MT-MinCutPool	Model Mean
Accuracy	train	0.646	0.916	0.659	0.688	–
	valid	0.69	0.687	0.692	0.69	–
	test	0.611	0.606	**0.639**	0.604	0.615
Sensitivity	train	0.618	0.931	0.766	0.76	–
	valid	0.675	0.715	0.78	0.756	–
	test	0.631	0.694	**0.777**	0.728	0.71
Specificity	train	0.676	0.901	0.544	0.612	–
	valid	0.704	0.658	0.597	0.619	–
	test	**0.594**	0.533	0.532	0.5	0.54
AUC	test	0.634	0.657	**0.692**	0.661	0.661

Table 5. Results for the proposed models.

		MTPool	GAF-Attention	MLSTM FCN	Model Mean
Accuracy	train	0.563	0.755	0.733	–
	valid	0.577	0.663	0.693	–
	test	0.517	**0.631**	0.586	0.578
Sensitivity	train	0.682	0.78	0.757	–
	valid	0.683	0.688	0.733	–
	test	0.655	0.622	**0.664**	0.647
Specificity	train	0.436	0.728	0.709	–
	valid	0.464	0.636	0.651	–
	test	0.4	**0.637**	0.521	0.519
AUC	test	0.538	**0.695**	0.648	0.627

Table 6. Results for the baseline models.

- The mean performance of the proposed models > baseline models
- **For proposed models:**
  - 🏆 GAF-CNN: Highest Accuracy, Sensitivity, AUC
  - 🥇 LSTM: Highest Specificity
- **For baseline models:**
  - 🏆 GAF-Attention: Highest Accuracy, Specificity, AUC
  - 🥇 MLSTM-FCN: Highest Sensitivity
- **For all models:**

Accuracy	0.639	GAF-CNN
Sensitivity	0.777	GAF-CNN
Specificity	0.637	GAF-Attention
AUC	0.695	GAF-Attention

Results and Discussion

		MT-MinCutPool	MTPool
Accuracy	train	0.688	0.563
	valid	0.69	0.577
	test	0.604	0.517
Sensitivity	train	0.76	0.682
	valid	0.756	0.683
	test	0.728	0.655
Specificity	train	0.612	0.436
	valid	0.619	0.464
	test	0.5	0.4
AUC	test	0.661	0.538

Table 7. Results for MT-MinCutPool and MTPool.

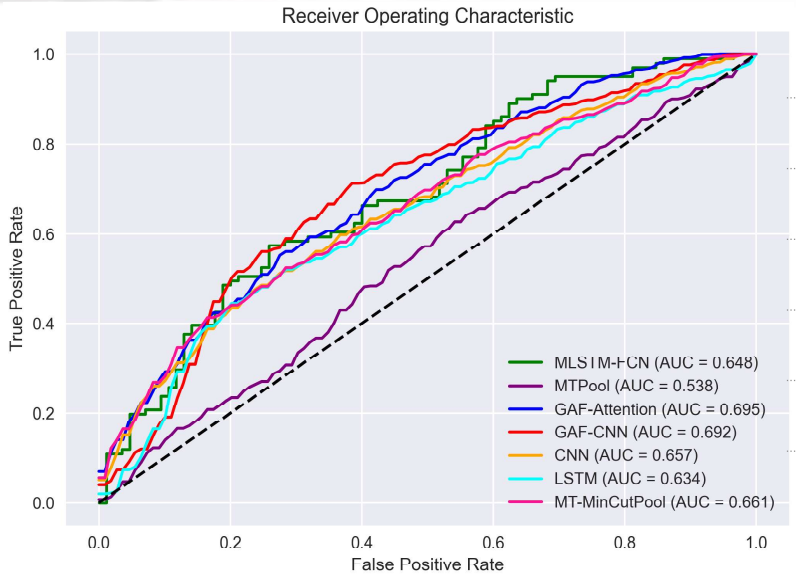
	MT-MinCutPool	**MTPool**
Graph Structure	Laplacian matrix	Correlation Coefficient matrix
Graph Pooling	MinCutPool	Adaptive Pooling

Table 8. Differences for MT-MinCutPool and MTPool.

- The performance of the MT-MinCutPool > MTPool for all metrics.

1. **Laplacian matrix:** Contain more graph information.
2. **MinCutPool:**
   - Directly identify strongly connected nodes, aggregate into one cluster.
   - Preserve important graph information during pooling process.

Results and Discussion



GAF-Attention	0.695
GAF-CNN	0.692
MT-MinCutPool	0.661
CNN	0.657
MLSTM-FCN	0.648
LSTM	0.634
MTPool	0.538

Figure 18. ROC curves for each model.

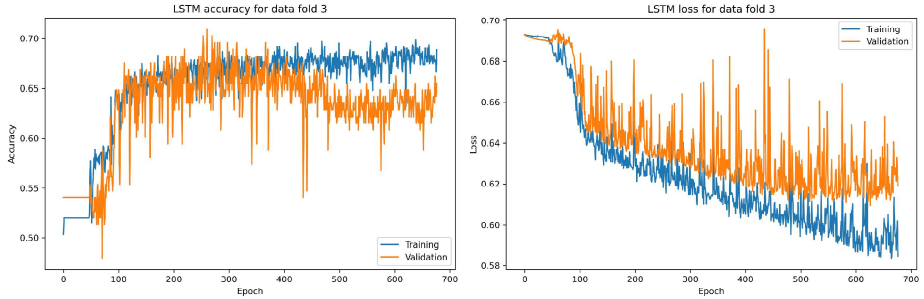


Figure 19. Accuracy and loss curves for LSTM model.

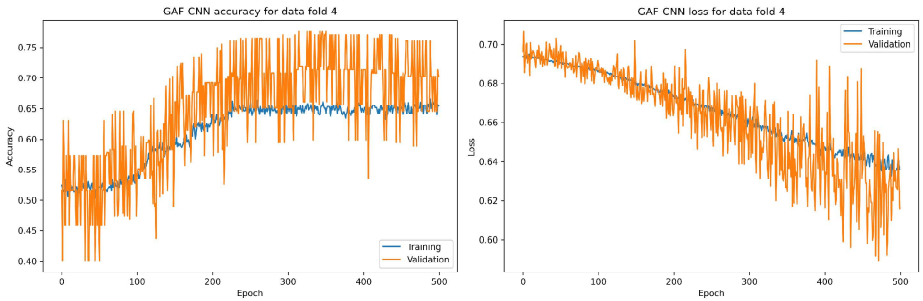


Figure 20. Accuracy and loss curves for GAF-CNN

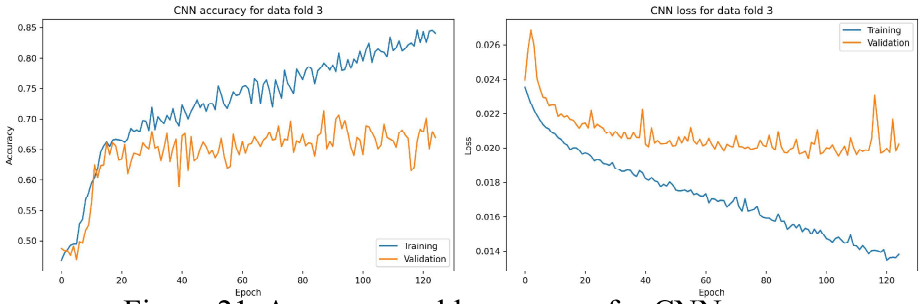


Figure 21. Accuracy and loss curves for CNN

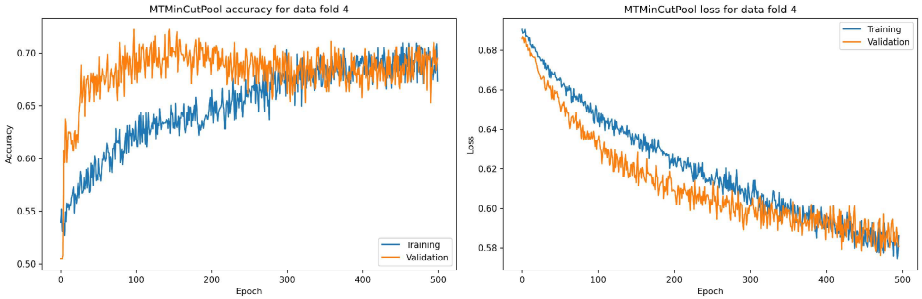


Figure 22. Accuracy and loss curves for MT-MinCutPool

Results and Discussion

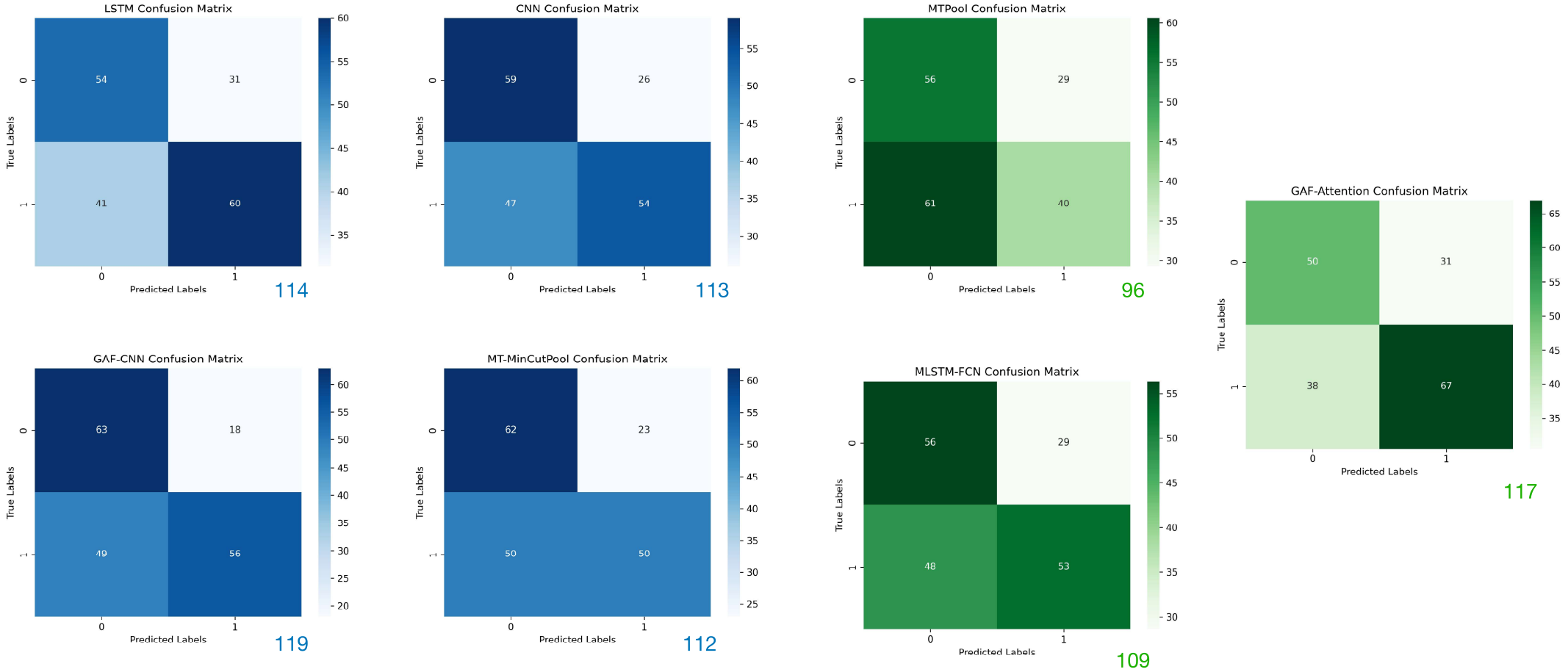


Figure 23. Confusion matrix for each model.

Contents

5. Other Attempt : Transfer Learning

Transfer Learning

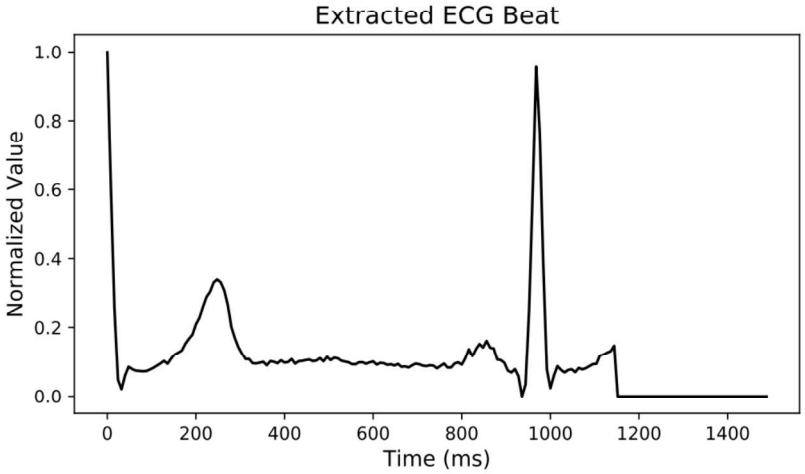


Figure 24. An example of an extracted ECG beat.

- Source data: ECG signals (PTB datasets)
- Source task: ECG heartbeat classification

Source and target data similarities:

- Time series data
- Binary classification tasks
- Normalized value ranges: (0, 1)

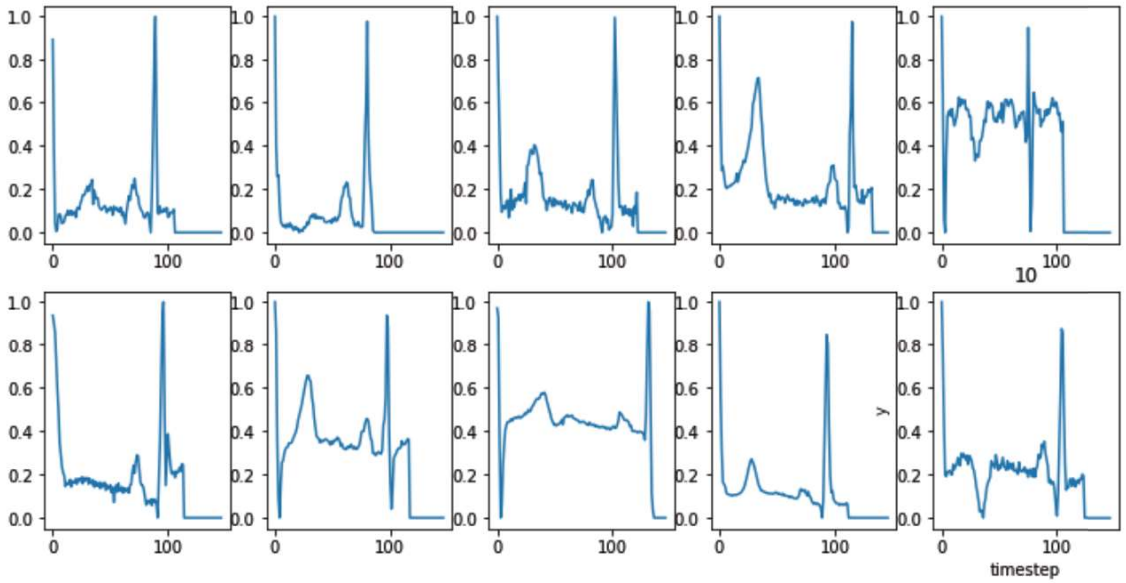


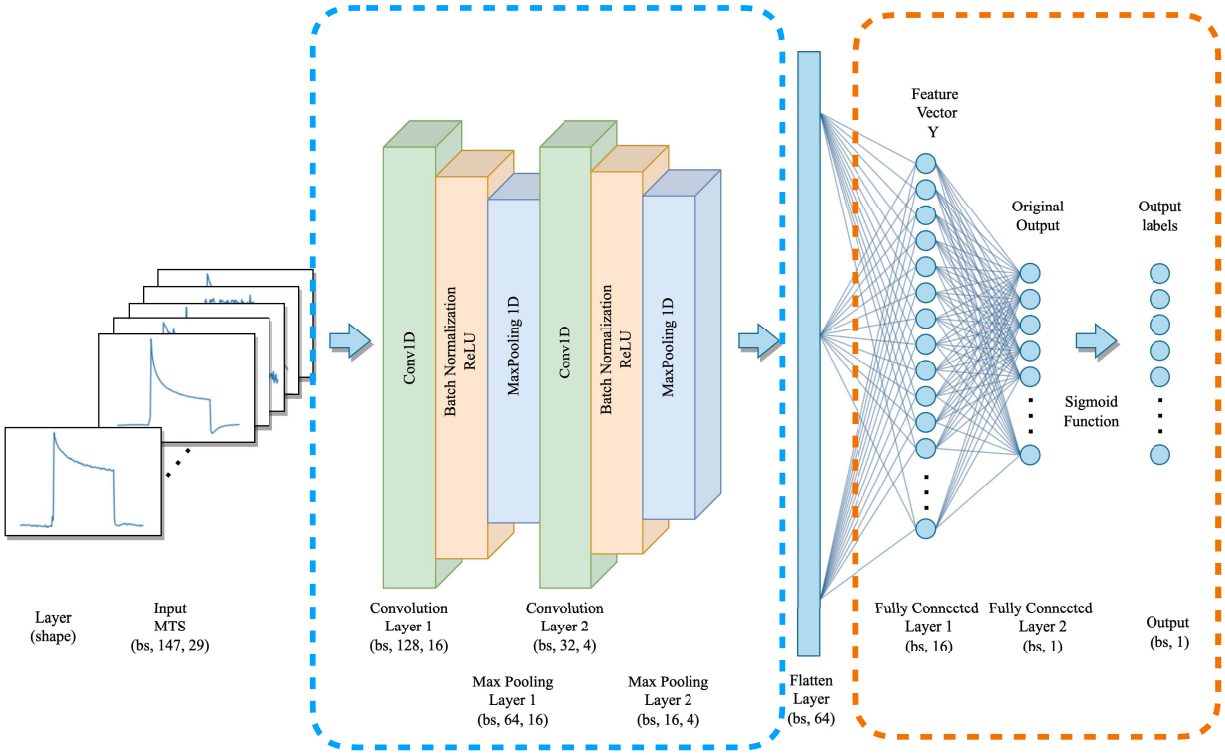
Figure 25. Different ECG signals.

Source and target data differences:

- Source: Univariate; Target: Multivariate
- Signal characteristics
- Number of source data (14552) >> Number of target data (928)

Reference: ECG Heartbeat Classification: A Deep Transferable Representation

Transfer Learning



- Train source model on source dataset.
- Freeze the first few layers of the pre-trained model.
- Train the pre-trained model on target dataset, update the last few layers.

Freeze the trainable parameters and weights from pre-trained model.

Update trainable parameters of new layers.

Results for Transfer Learning

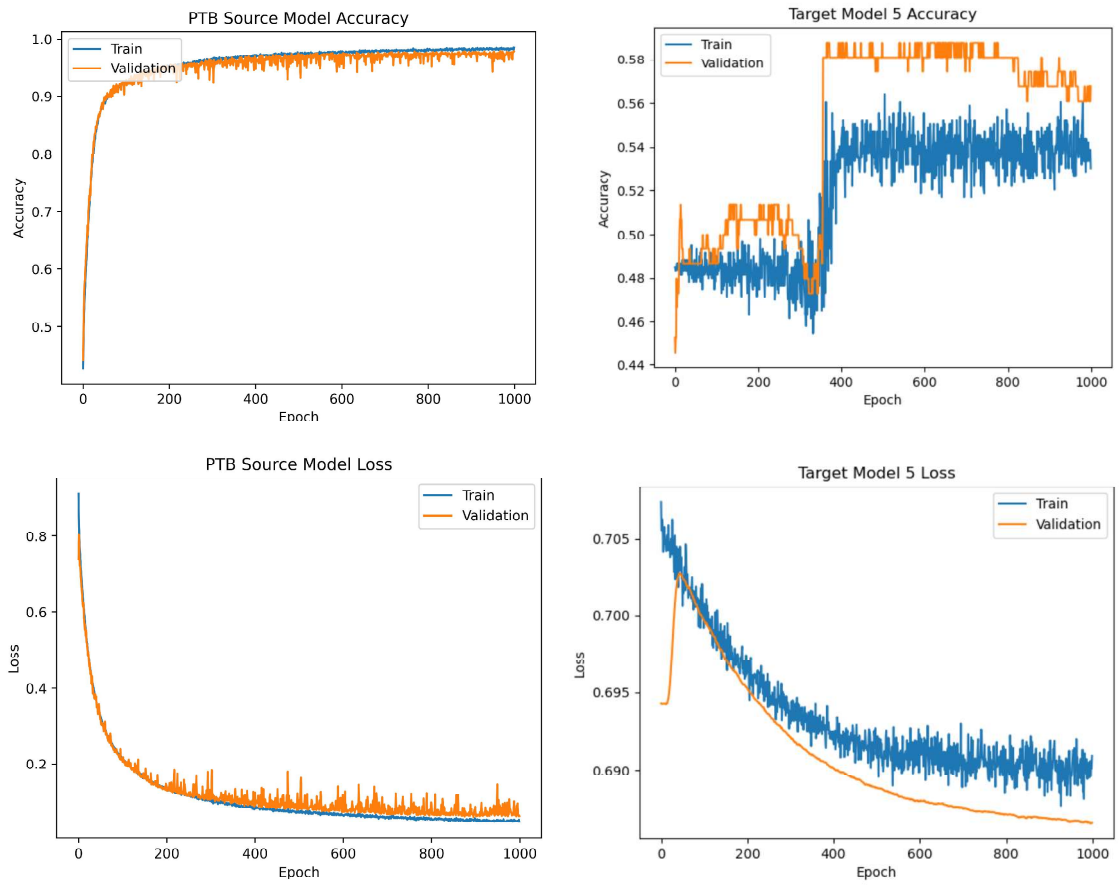


Figure 26. Accuracy and loss curves of source data (left), and target data (right).

		TL target	CNN
Accuracy	Train	0.528	0.916
	Valid	0.582	0.687
	Test	0.514	0.606
Sensitivity	Train	0.75	0.931
	Valid	0.817	0.715
	Test	0.804	0.694
Specificity	Train	0.293	0.901
	Valid	0.333	0.658
	Test	0.27	0.533
AUC	Test	0.539	0.657

Table 9. Evaluation metrics for TL target and CNN.

- High sensitivity $= \dfrac{TP}{TP + FN}$
- Low accuracy, specificity and AUC score
- **Domain shift**

Contents

6. Conclusion and Future Research

Conclusion and Future Research

TECHNION Israel Institute of Technology

Summary

1. Data Preprocessing
- Middle Part Extraction
- Data Normalization
- Data Calibration
- Good Sensor Selection

2.1 Proposed ML Models
- LSTM
- CNN
- GAF-CNN
- MT-MinCutPool

2.2 Baseline ML Models
- MLSTM-FCN
- GAF-Attention
- MTPool

3. Model Training
- 20% test
- 80% train+validation
- 5 fold cross validation
- 10 runs

4. Model Evaluation
- Accuracy
- Sensitivity
- Specificity
- ROC-AUC

5. Result Discussion

Conclusion and Future Research

TECHNION Israel Institute of Technology

Conclusion

- Proposed models exhibits overall better performance than baseline models.
  - MT-MinCutPool outperformed MTPool in all aspects.
  - GAF-CNN has highest accuracy and sensitivity among all models.
- Lightweight models performed better than complex models for small dataset problems.

Limitations

- The focus on a single TB dataset
- The small size of the TB dataset

Conclusion and Future Research

TECHNION Israel Institute of Technology

Future Research

- Explore the efficacy of the proposed models on other types of TB datasets / similar MTSC problems/ larger datasets.
- Explore the potential of transfer learning (TL) if a more extensive and similar source dataset could be found.
- Explore surrogate data approaches, such as using Variational Auto Encoder (VAE).
- Explore the interpretability of the proposed models in the context of TB diagnosis.

Contents

TECHNION
Israel Institute
of Technology

Questions?

Acknowledgements

TECHNION
Israel Institute
of Technology

Thank You ! 😄