DS 303 PROJECT

MAY 2, 2023

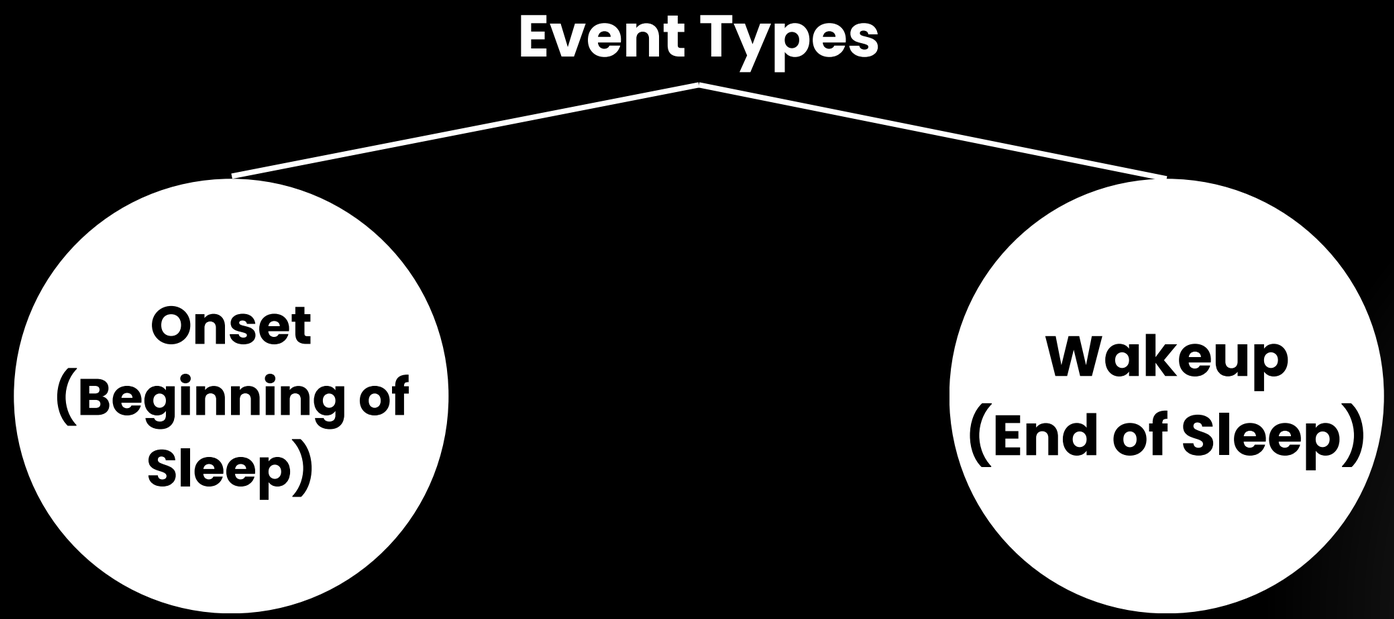
SLEEP STATE DETECTION

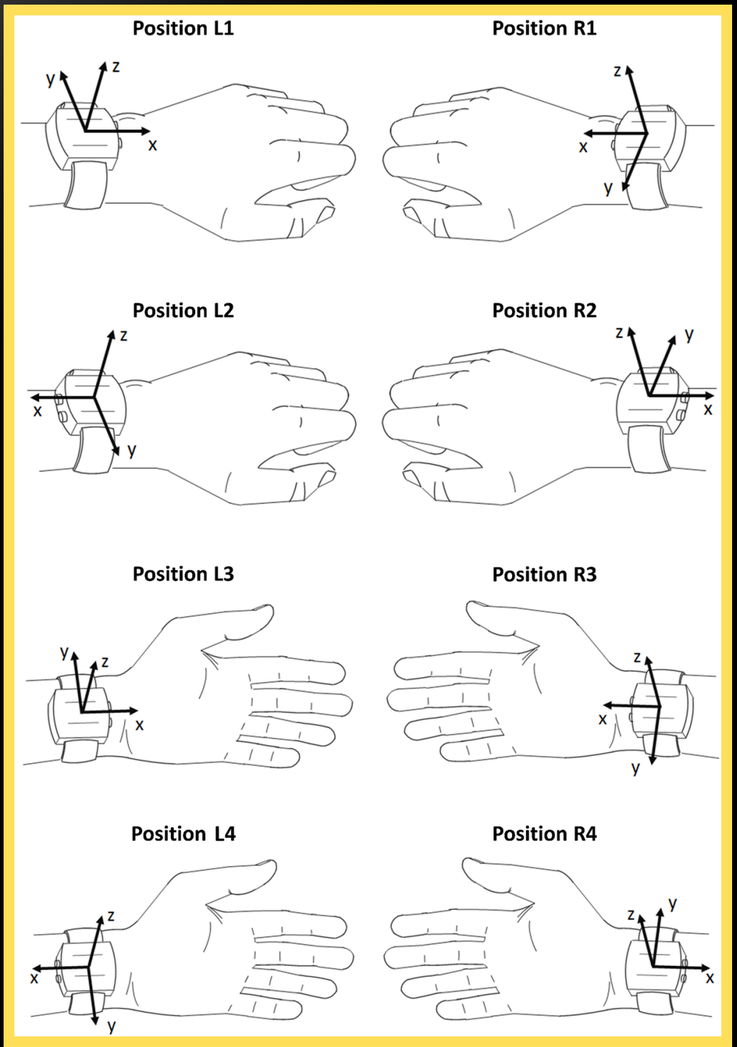
TEAM 1

SHUBHAM SHARMA (23B0312)
RAHUL(23B0305)
KSHITIJ RAJ SURYAVANSHI (23B0441)
JAYANT DESHMUKH (23B0431)
RISHIT KESHARWANI(23B0439)

ABOUT

The idea of the project is to use recordings of wrist-worn accelerometer data annotated with **two event types**: ***onset***, the beginning of sleep, and ***wakeup***, the end of sleep, and employ it to detect the occurrence of these two events in the accelerometer series.





WHY STUDY THIS?

- Wearable accelerometer data offers a scalable, **non-invasive alternative** to traditional sleep logging.

- It can help advance digital health solutions by enabling accurate monitoring of sleep patterns in real-world, **long-duration settings**.

WHAT WE AIMED FOR

Using the dataset, we wish to do the following:

Detect Sleep Events

Identify the onset (start) and wake-up (end) of sleep periods from continuous accelerometer signals.

Train a Model

Build and train models to automatically detect sleep boundaries across multiple subjects and days.

DATASET

Files in the dataset:

train_series.parquet
Multiday wrist accelerometer data for each subject

test_series.parquet
Same format as training data, used for predictions

train_events.csv
Annotated sleep events (onset, wakeup) for training series.

DATASET

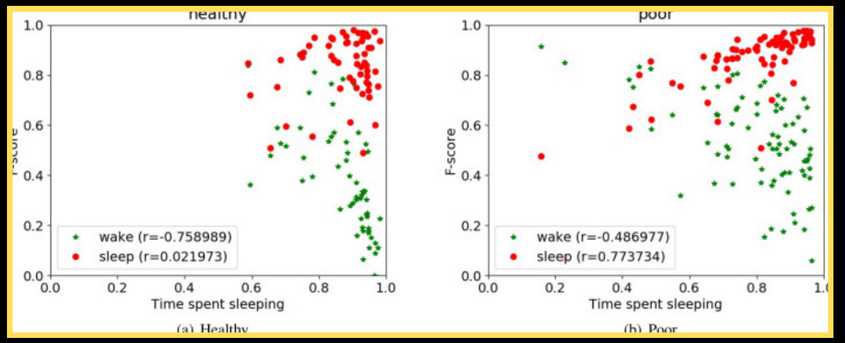
Feature	Description
series_id	Unique ID for each subject's continuous recording
step	Timestep index within a series
timestamp	Datetime of each observation (ISO 8601 format)
anglez	Z-angle of the wrist (arm posture, used in sleep detection)
enmo	Movement intensity (Euclidean Norm Minus One)
event	Sleep onset or wakeup (only in train_events.csv)

RELATED WORKS

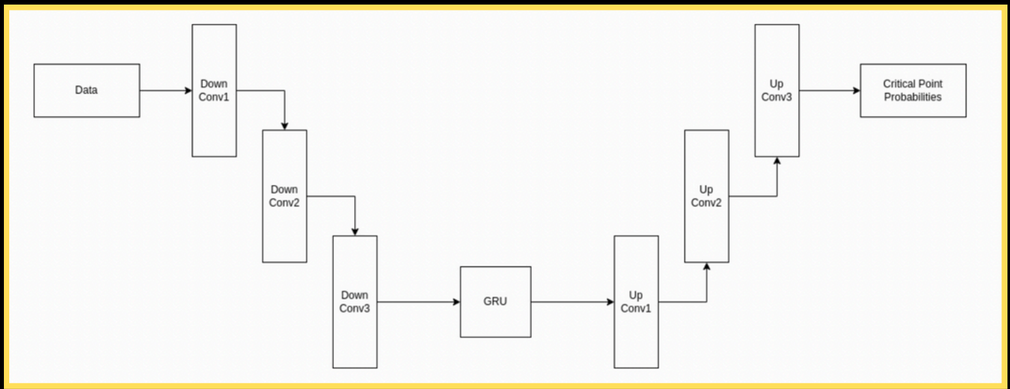
$$\max\{(\sqrt{acc_x^2 + acc_y^2 + acc_z^2} - 1), 0\}$$

- van Hees et al. (2013) developed a method to detect sleep using wrist-worn accelerometer data by introducing the **ENMO (Euclidean Norm Minus One) metric**, enabling sleep parameter estimation without requiring subjective sleep diaries.

- Rahman et al. (2020) evaluated traditional machine learning models like **Random Forests, SVMs, and k-NN** for sleep detection using wrist-worn accelerometer data, showing reasonable accuracy but highlighting limitations in generalising across subjects and handling noisy or irregular activity patterns.

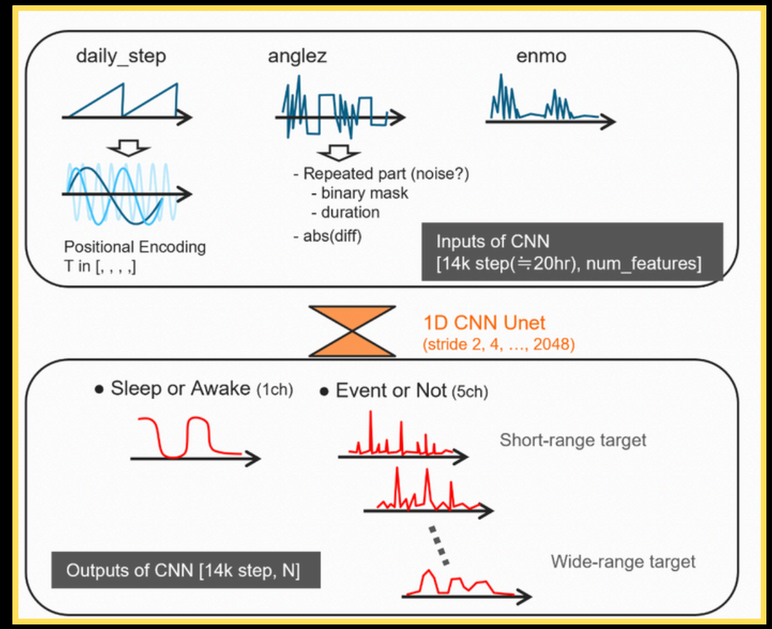


RELATED WORKS

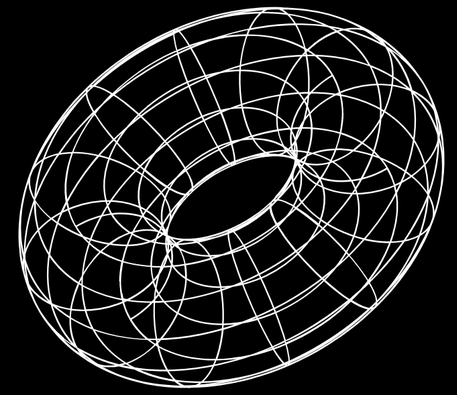


- The first place solution on Kaggle has a model with a structure comprising: **CNN (down sample) → Residual GRU → CNN (up sample)**

- The second place solution pipeline leverages a **1D-CNN (U-Net)** for event detection and classification, while **LGBM** is used for rescoring predictions and optimising the final event accuracy curve.



DATA PREPROCESSING



Normalization

Applied z-score normalization per axis (x, y, z)

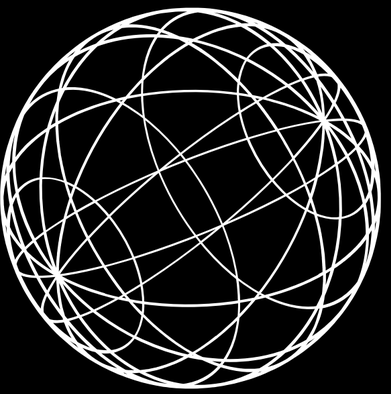
Truncation

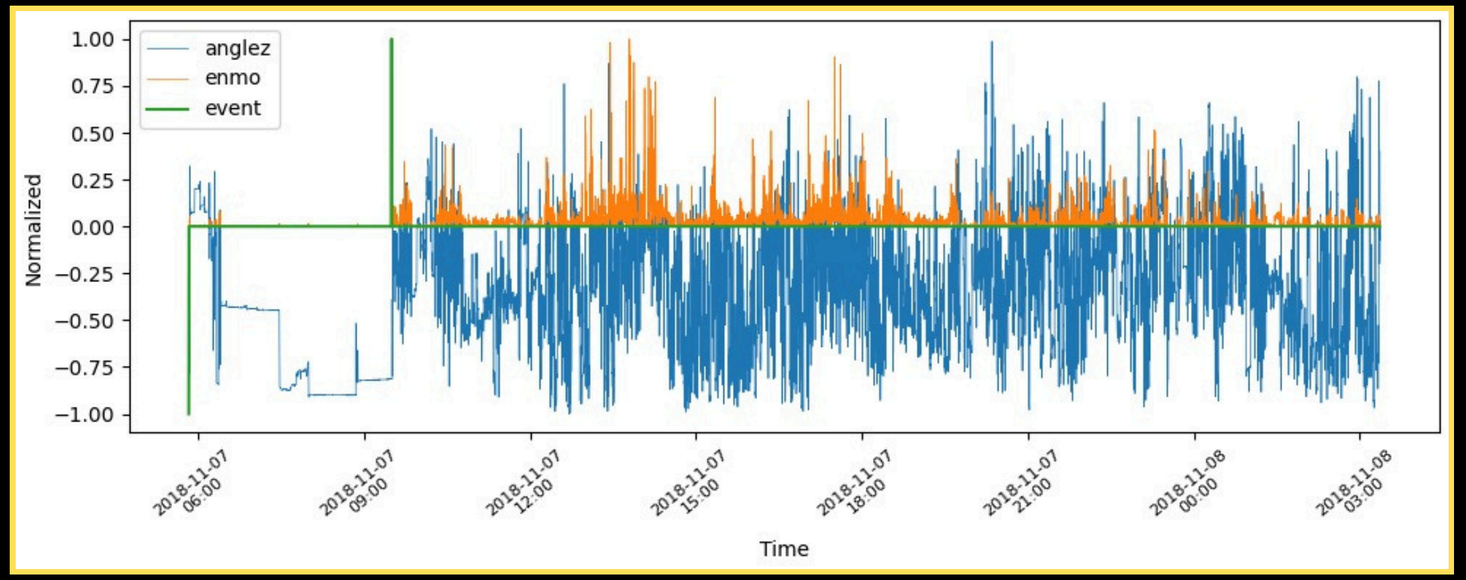
Truncation of data series around onset and wake-up events

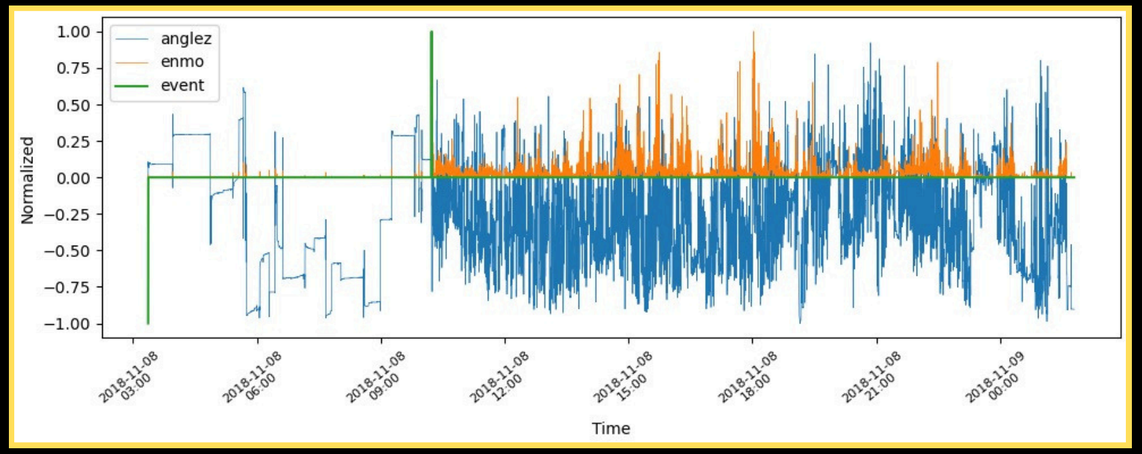
Tolerance

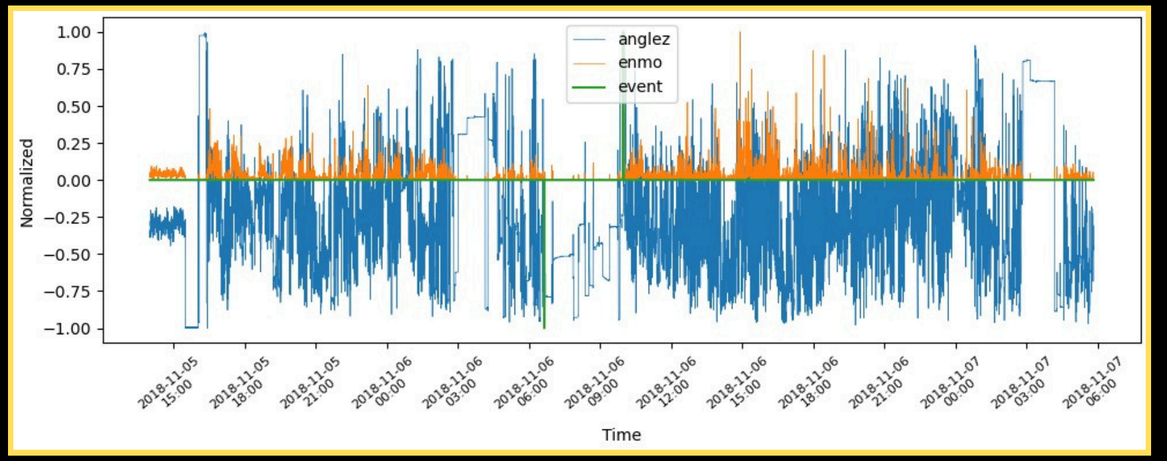
Too many samples of no events removed, took around 1000 steps around onset and wakeup events

Applied hard tolerance labelling around the events to address class imbalance




anglez
enmo
event
Normalized
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
2018-11-07 06:00
2018-11-07 09:00
2018-11-07 12:00
2018-11-07 15:00
2018-11-07 18:00
2018-11-07 21:00
2018-11-08 00:00
2018-11-08 03:00
Time


anglez
enmo
event
Normalized
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
2018-11-08 03:00
2018-11-08 06:00
2018-11-08 09:00
2018-11-08 12:00
2018-11-08 15:00
2018-11-08 18:00
2018-11-08 21:00
2018-11-09 00:00
Time


anglez
enmo
event
Normalized
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
2018-11-05 15:00
2018-11-05 18:00
2018-11-05 21:00
2018-11-06 00:00
2018-11-06 03:00
2018-11-06 06:00
2018-11-06 09:00
2018-11-06 12:00
2018-11-06 15:00
2018-11-06 18:00
2018-11-06 21:00
2018-11-07 00:00
2018-11-07 03:00
2018-11-07 06:00
Time

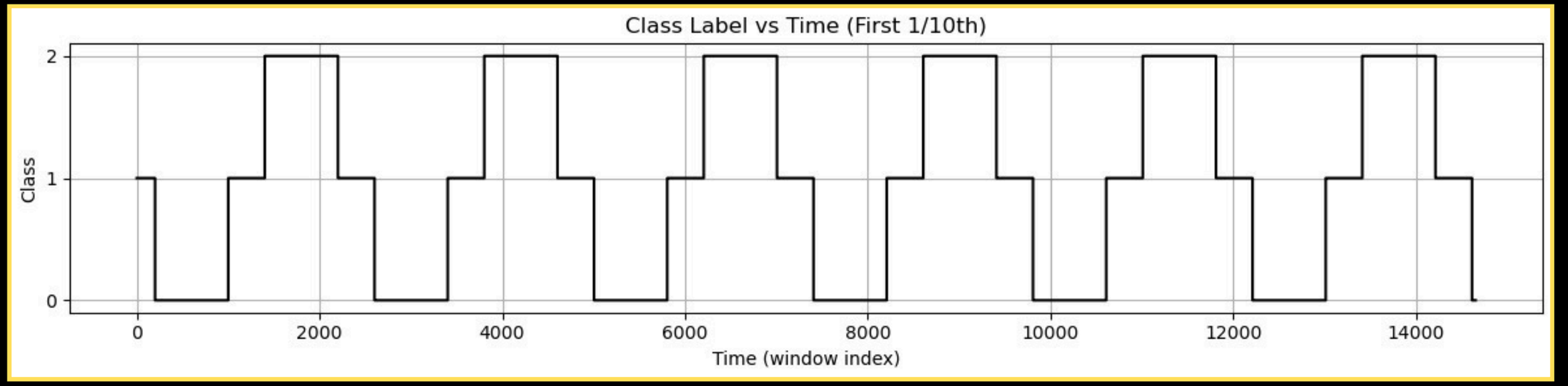
DATA PREPROCESSING

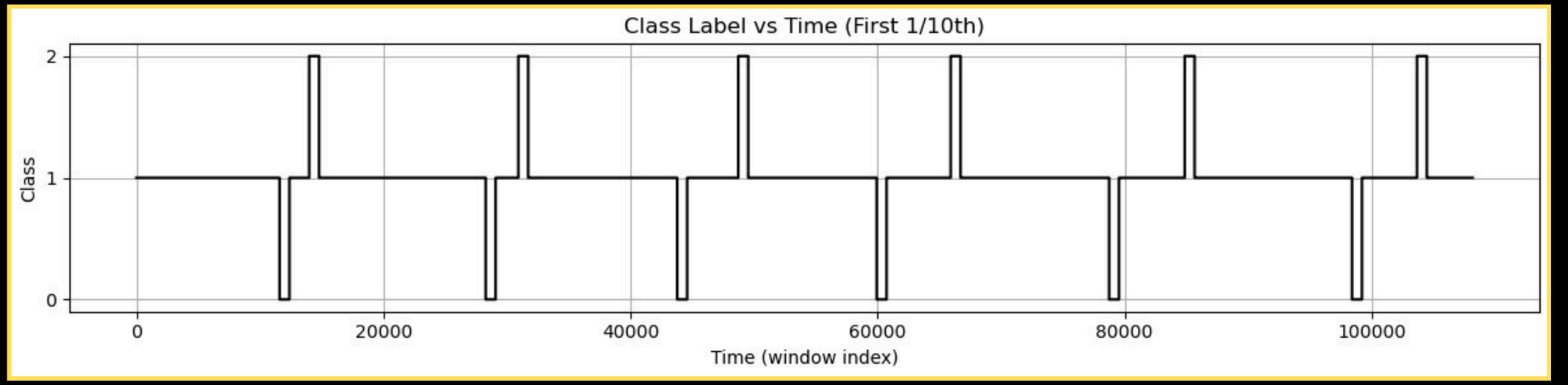
Label Assignment:

Assigned majority labels within each window from the train_events.csv.
Assigned labels as follows- onset 0, no event 1 and wake-up 2

Missing Values & Noise:

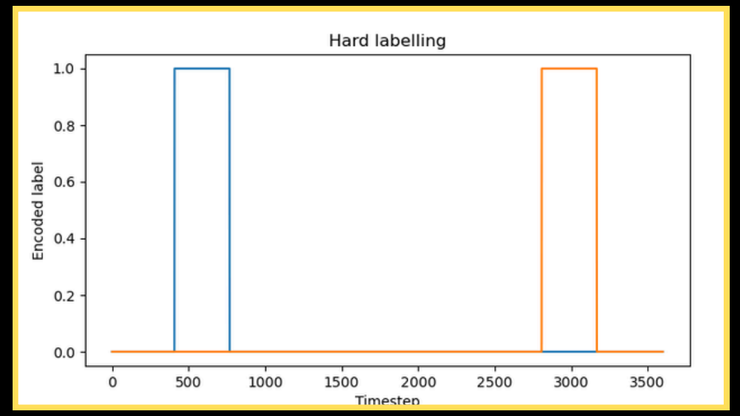
Checked for gaps and handled missing timestamps via forward-fill or interpolation.
Low variance/noise segments were filtered out to reduce redundancy.


Class Label vs Time (First 1/10th)
Class
2
1
0
0
2000
4000
6000
8000
10000
12000
14000
Time (window index)

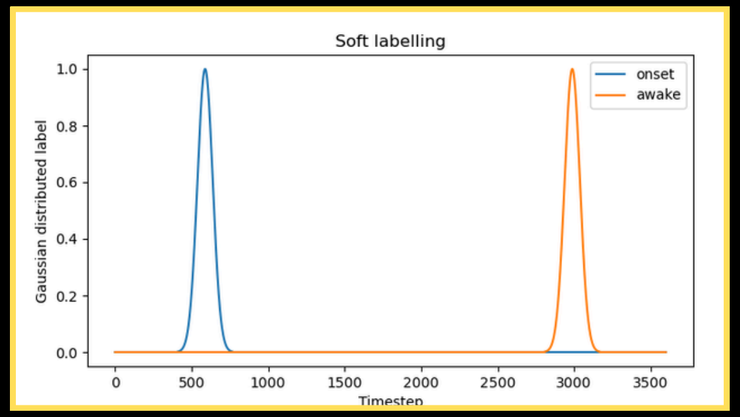

Class Label vs Time (First 1/10th)
Class
2
1
0
0
20000
40000
60000
80000
100000
Time (window index)

APPROACH

We tried 2 approaches for preparing the labels in this problem



1.Hard labelling is created by pre-selecting the number of offset from the events and considering it as a part of the training labels.



2. Soft labelling is created usia ng Gaussian distribution around the onset/awake positions with different sigma and offset values

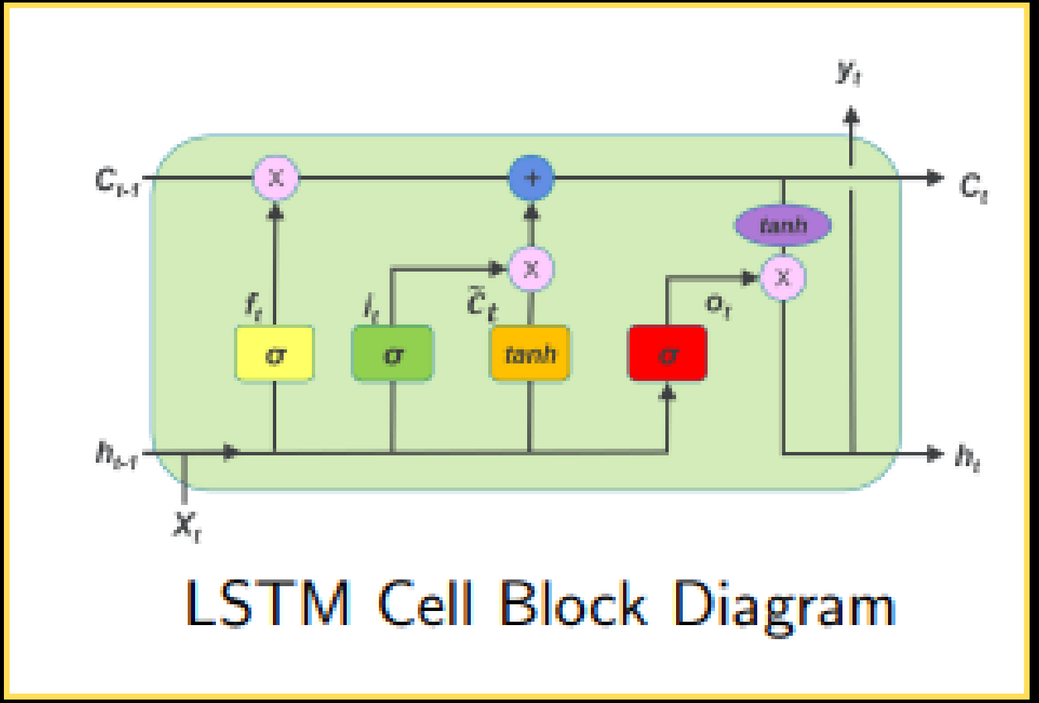
MODEL ARCHITECTURE

BiLSTMEventDetection

→ Architecture
- BiLSTM (captures past ↔ future context)
- Hidden size: 128, Layers: 1
- FC head → 3-class outputs (onset / no-event / wake)

→ Training
- Loss: Cross-Entropy
- Optimizer: Adam

→ Data
- 80/20 split (train / val)
- PyTorch DataLoader for batching & shuffling



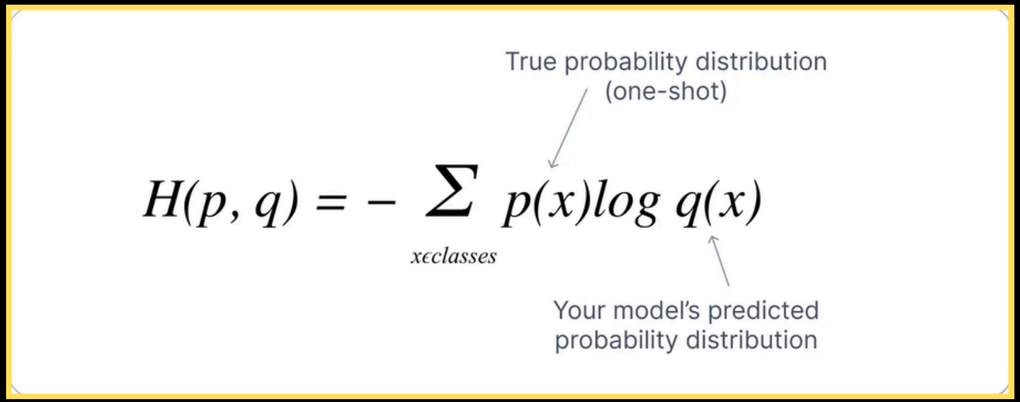
LSTM Cell Block Diagram

LOSS & OPTIMISATION

Loss Function:
Cross-Entropy Loss chosen for probabilistic binary classification. Alternatives like MSE (less suited for classification) and Focal Loss (for heavy imbalance) were considered but not adopted due to manageable imbalance.

Optimisation Strategy:
Optimiser: Adam — robust for time-series and sparse gradients.
Learning rate: Cosine annealing schedulers adjust the learning rate following a cosine curve, gradually reducing the rate over each cycle.
CoTuned starting from 1e-3, with ReduceLROnPlateau to adjust during training.

Regularization:
Dropout layers, early stopping, and batch normalization to combat overfitting.



EVALUATION

Training Setup:

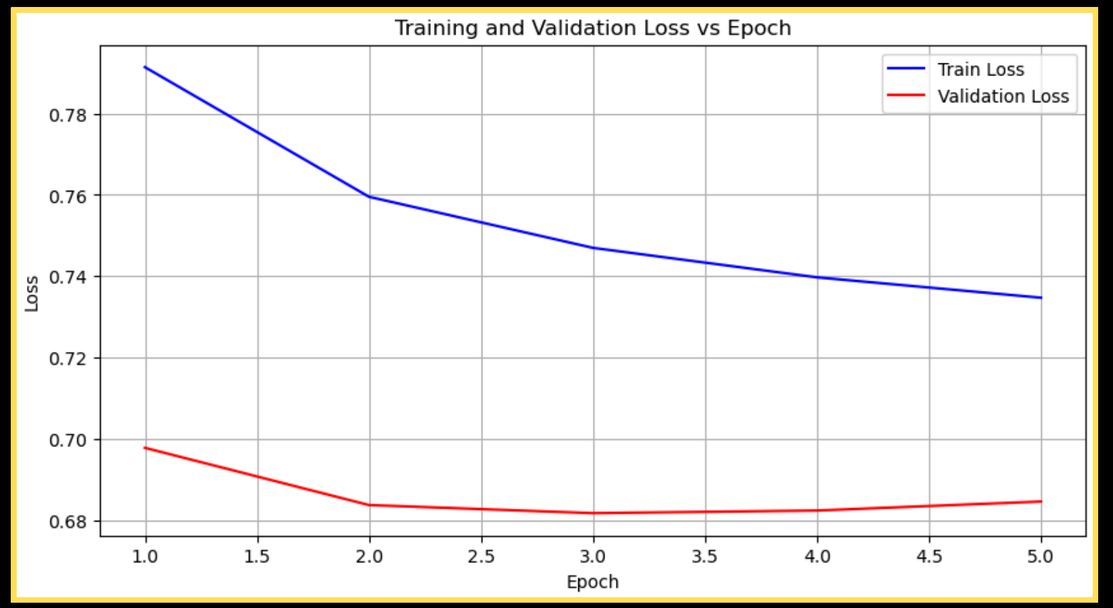
Train/Validation/Test Split stratified by series_id.
Batch size: e.g., 64; Epochs: 50–100; Callbacks: Early Stopping, LR Scheduler.

Metrics:

Used F1-score, Precision, Recall, and Accuracy to address label imbalance.

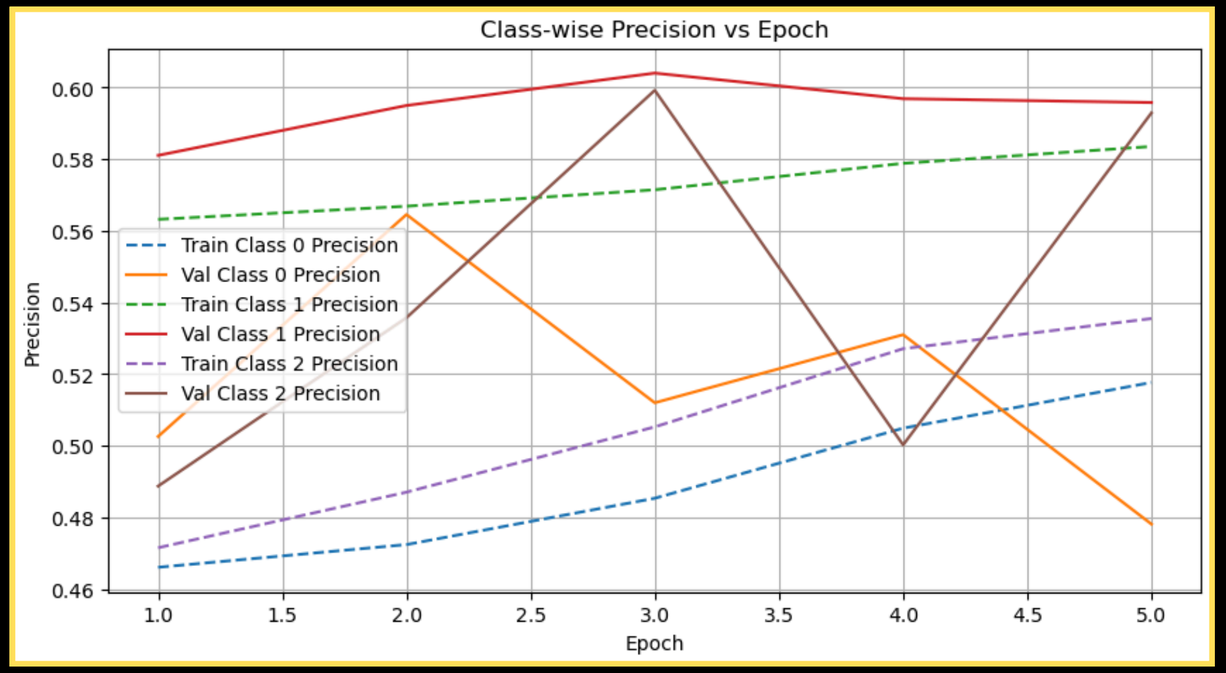
RESULTS

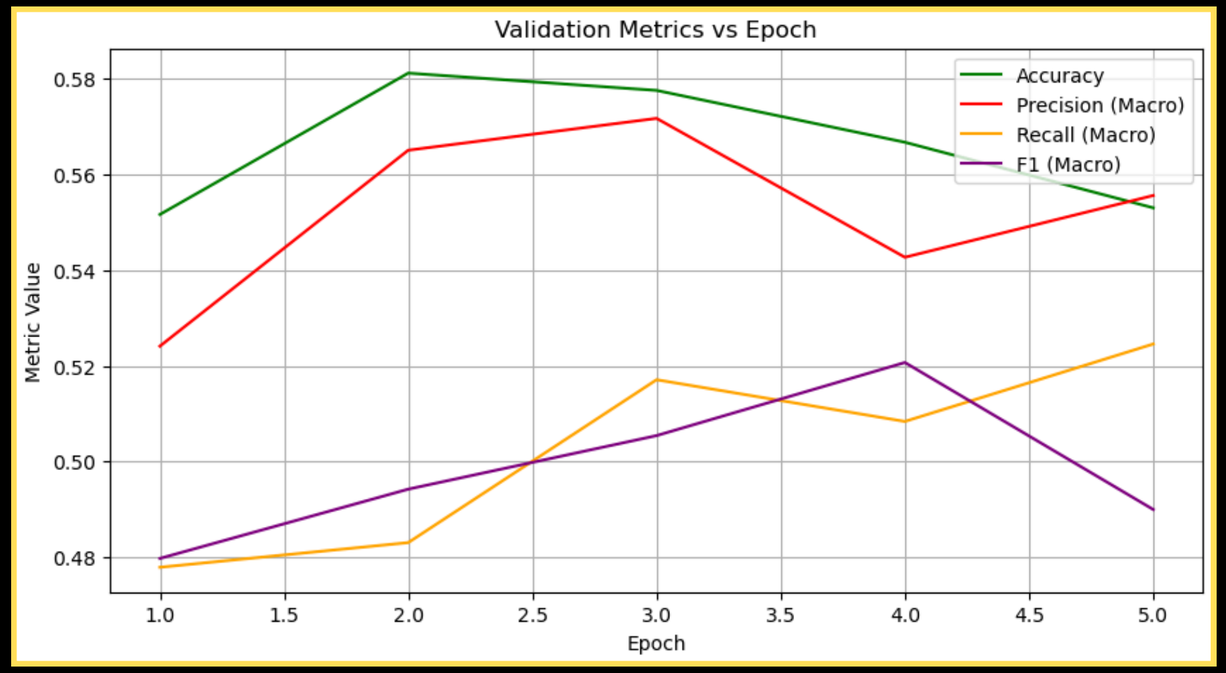
Class	Precision	Recall	F1 Score
0	0.5146	0.4756	0.4943
1	0.591	0.7548	0.6629
2	0.5643	0.2168	0.3132



1. We can see training loss goes down with the validation loss up to the third epoch after that we can see some overfitting due to our limited data

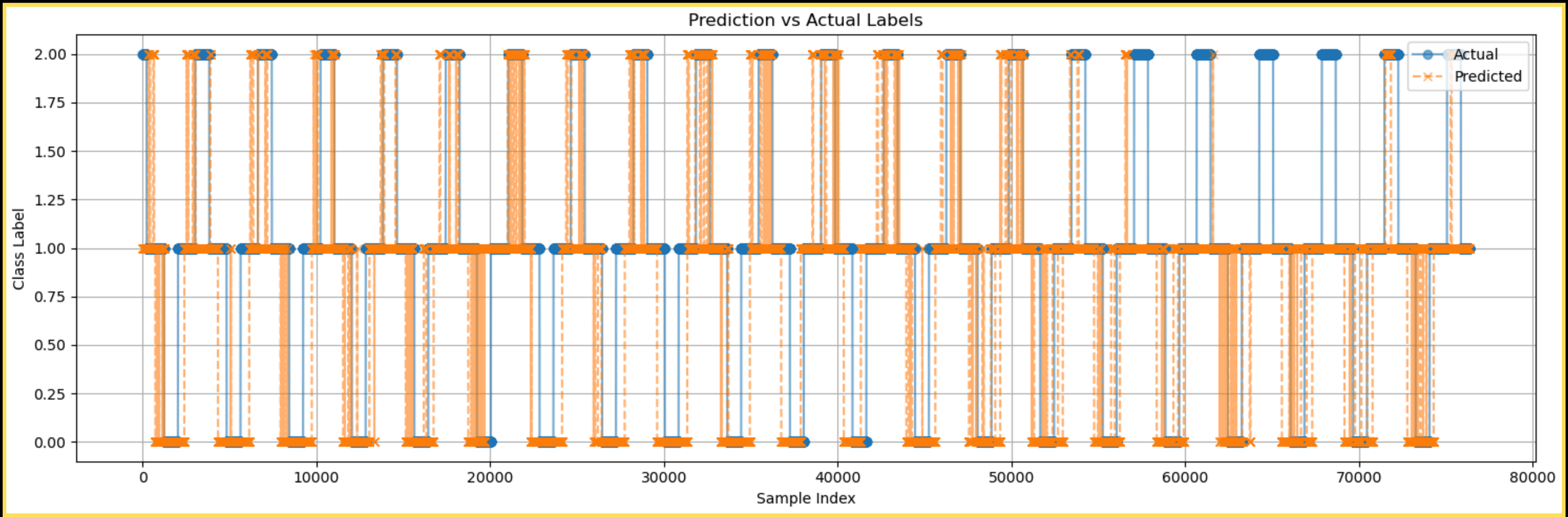
RESULTS





2. We get a relatively decent score of all three Metrics considering the problem complexity and the limited computational resources we have

RESULTS



3. While the evaluation metrics suggest the model's overall performance is modest, a comparison between the actual and predicted labels reveals that the model effectively captures the underlying pattern and temporal structure of the true class sequence.

THANK YOU