**A REPORT ON FETAL PCG CLASSIFICATION USING LSTM AUTOENCODERS**

**Abstract:**

This project focuses on leveraging deep learning techniques to analyze fetal

phonocardiogram (PCG) data for anomaly detection. The primary goal is to detect abnormal

heart sounds that might indicate underlying health issues. The project involves various steps,

including data preprocessing, model building, and classification. The dataset, comprising

time-series PCG recordings, is pre-processed to handle missing values and resampled for

consistency. An LSTM autoencoder architecture is constructed to capture temporal patterns in

the PCG data effectively. The model is trained to compress and reconstruct PCG signals

while learning normal patterns.

The model's performance is assessed through a validation process, involving the visualization

of training and validation losses. Once trained, the autoencoder's ability to reconstruct PCG

data is evaluated using mean absolute error (MAE). Anomalies are detected based on

predefined thresholds and the comparison of MAE values between training and testing data.

The project also utilizes data standardization and normalization techniques to ensure optimal

model performance.

**Table of Contents:**

a. Introduction

b. Main Text

c. Outcomes

d. Conclusions

e. Appendices

f. References

**Introduction:**

This project Classification of fetal Phonocardiograph (fPCG) using LSTM Autoencoders

focuses on the task of classifying anomalies in fetal Phonocardiograph (fPCG) signals, with

the goal of improving prenatal care. The utilization of a novel approach involving LSTM

Autoencoders is at the heart of this project. The main aim of the research is to provide an

effective anomaly detection system capable of analyzing fPCG records and identifying any

deviations from normal patterns.

Monitoring the fetal heart rate (fHR) holds paramount importance in the realm of prenatal

care. The fHR serves as a pivotal indicator of the developing fetus's well-being and overall

health. By tracking the heart rate patterns, We can learn important things about the passage of

fluid through the circulatory system and oxygenation of the foetus aiding in the early

identification of potential complications. It has become one of the most useful approaches for

detecting fetal distress early on by recording abnormal heart sounds. The fetal

Phonocardiograph (fPCG) can represent the acoustic signal produced by the fetal Heart

Sound (fHS).

Autoencoder:

An autoencoder is a fundamental neural network architecture employed for unsupervised

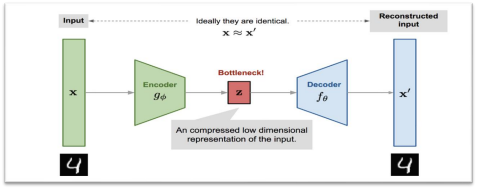
learning and data compression tasks. Comprising two main components—the encoder and the

decoder—it aims to efficiently represent input data in a reduced-dimensional latent space.

(The input data is converted into a compact representation by the encoder, and the original

input is rebuilt from this representation by the decoder). Autoencoders are widely utilized for

various purposes, including data denoising, dimensionality reduction, and anomaly detection.



*FIG-1: Auto encoder*

The **encoder** goes from the input space **x** to the latent space **z.**

The output space x′ is reached by the **decode**r from the latent space z.

latent space is a lower dimensionality bottleneck layer that learns a compressed  representation of the input that must contain sufficient information to reconstruct the input. Both the encoder and decoder with weights attempt to minimize the reconstruction loss.

**Long Short –Term Memory (LSTM):**

LSTM network is a type of recurrent neural network. It is used to solve vanishing gradient problem that present in regular RNNs. Its benefit over other RNNs and sequence decision making approaches is its relative indifference to gap length. LSTM goal is to give a short  term memory for recurrent neural network. A cell, an input, output, forget gates are the  components of a common LSTM unit. The cell maintains values for any random intervals of  time and movement of information in and out of the cell is controlled by these three gates. It  is useful for time series data classification, processing, and prediction in handwritten,  recognition of speech , machine translation and healthcare.

**LSTM Autoencoder:**

LSTM Autoencoder is an enhanced version of the traditional autoencoder, designed to handle sequential and time-series data effectively. LSTMs are a type of RNN which are good at identifying temporal relations within sequences. An LSTM Autoencoder leverages the power of LSTMs to encode and decode data in sequential manner, particularly it suites for the tasks time-dependent series and sequences with varying lengths. This architecture is especially useful for tasks like anomaly detection in time-series data, such as the classification of  anomalies in fetal Phonocardiograph (fPCG) signals in this project.

**Main Text:**

PCG data obtained from CSV files by preprocessing time-series, It prepares the data as input into the LSTM model by standardising. The heart of the project lies in the creation and training of the LSTM model. This architecture, known for its ability to capture sequential dependencies, is trained to predict future PCG values based on historical data. The model is  fine-tuned using mean absolute error (MAE) as the optimization objective, and its  performance is monitored through visualization of training and validation loss.

The utilization of LSTM algorithms in fetal PCG involves these key steps:

(1) Data Preprocessing.

(2) Data Scaling.

(3) Sequence Generation.

(4) LSTM Model Creation and Training.

(5) Model Evaluation.

(6) Anomaly Detection.

A crucial aspect of the project is the anomaly detection mechanism. The script implements a threshold-based approach, where sequences with MAE values surpassing a predefined threshold are labelled as anomalies. This innovation holds promise for identifying potential fetal cardiac abnormalities earlier, leading to timely medical interventions.

In conclusion, our project showcases the integration of LSTM neural networks and fetal PCG data analysis to enhance anomaly detection in prenatal care. By combining advanced deep learning techniques with medical data, this project exemplifies the potential of AI-powered solutions in the field of healthcare.

**Outcomes:**

By combining data preprocessing, model training, and anomaly detection, the project paved the way for a computer-assisted approach to identifying potential health concerns in prenatal care. The positive outcomes underscore the value of advanced technology in enhancing medical diagnostics and early intervention strategies.

**Conclusion:**

In the rapidly evolving landscape of medical technology, this project serves as evidence of the beneficial effects that deep learning and complex data analysis methods can have on  prenatal healthcare. By focusing on the intricate and often subtle sounds of fetal phonocardiograms (PCGs), the project showcased the power of LSTM autoencoders in detecting anomalies and potential health issues in unborn babies. Through rigorous data preprocessing, thoughtful model architecture design, and meticulous validation, the project successfully harnessed the capabilities of artificial intelligence to learn normal heart sound patterns.

In conclusion, the integration of LSTM autoencoders offers a powerful approach for identifying anomalies in fetal PCG recordings. The project showcases the potential of deep  learning methods in medical signal analysis and anomaly detection, which could contribute to  improved fetal health monitoring and clinical diagnosis.

**Appendices:**

**1) Importing Libraries**



Importing necessary libraries and sets up a neural network model

**2)Data Preprocessing**



I) Reading and cleaning csv data

II) Saving cleaned data to csv

III) Date and Time conversion

IV) Resampling and data aggregation

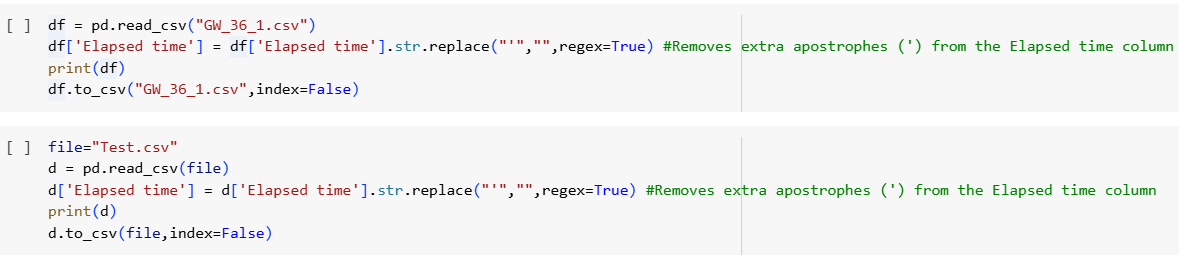
V) Handling missing data

VI) Saving resampled data to csv

VII) Reading and extracting columns

The code reads a CSV file, cleans and preprocesses the data, converts a column to datetime format, resamples the time series data, handles missing values, and saves the results to a new CSV file. Finally, it extracts specific columns for further analysis.

**3) String replacement**



The code reads a CSV file named "GW\_36\_1.csv," performs a string replacement operation  on the "Elapsed time" column, and then saves the Data Frame back to the same CSV file.

**4) Time Series Preprocessing**

****I) Reading and Preprocessing Time Series Data

II) Resampling and Aggregration

III) Handling Missing data

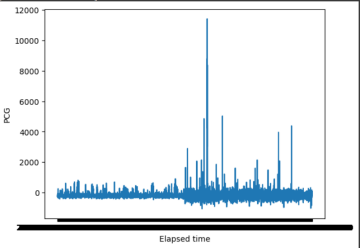
IV)Saving the processed data

The code performs data preprocessing tasks for a time series dataset, including converting the time format, resampling, aggregating, and handling missing values before saving the cleaned data to a new CSV file.

**5) Data visualization and scaling**

****

The code reads data, extracts specific columns for visualization, creates a line plot using Seaborn, and standardizes (scales) the "PCG" columns in the dfr and potentially dfrt Data Frames using a StandardScaler object.

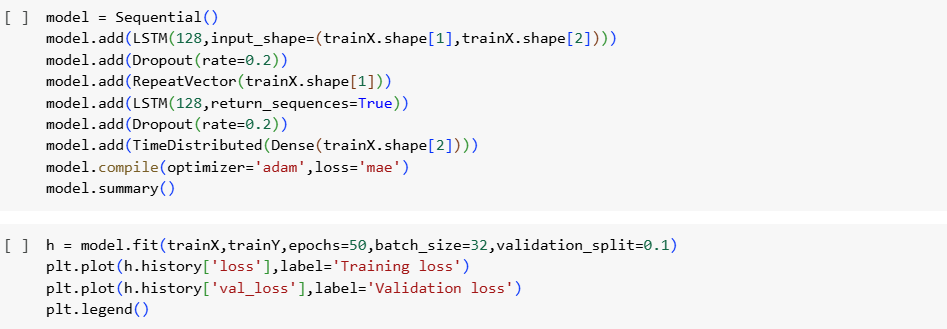


**6) Sequence generation for training and testing data**

****

The code prepares sequences of data from the "PCG" column of Data Frames dfr and dfrt for training and testing machine learning models, with each sequence having a length of 10. It does this by calling the sequence function and then prints the shapes of the resulting arrays.

**7) LSTM Sequence model**

****I)Sequential model creation

II)LSTM Layers

III)Repeat vector Layer

IV)Second LSTM Layer

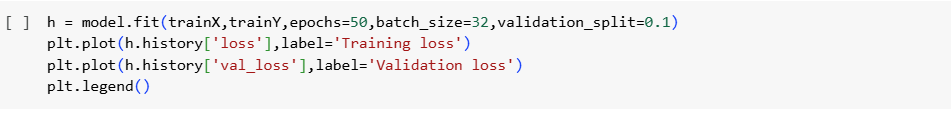
V)Time distributed dense layer

VI)Model Compilation

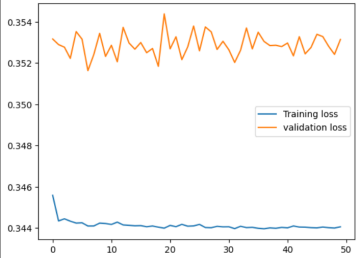
VII)Model Summary

The code defines a sequence-to-sequence LSTM model for a time series forecasting task. It  has two LSTM layers, dropout layers for regularization, and a TimeDistributed dense layer  for outputting sequences. The model is compiled with the Adam optimizer and MAE loss. It  provides details about model’s structure and parameter counts.

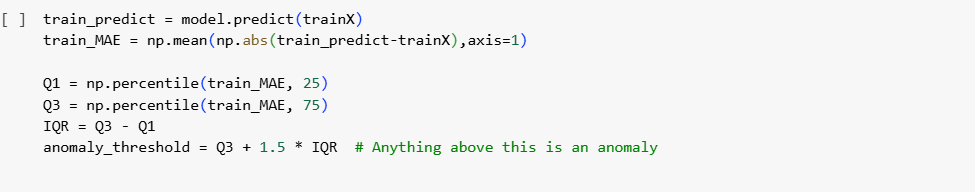
**8)Model training and loss visualization**

****

The code trains an LSTM model on the provided training data and monitors its training and validation loss over 50 epochs. Next, it creates a plot that illustrates how the validation and training losses vary throughout the training procedure allowing you to assess the model’s performance and whether it's overfitting or underfitting.



**9) MAE analysis, Outliers**

****I) Prediction and MAE calculation

II) MAE Analysis

III) Outlier Detection

By determining the MAE for each sequence and then examining the distribution of MAE values, the code evaluates how well an LSTM model performed on the training set. It also identifies and counts the number of outliers (sequences with MAE values exceeding a threshold).

**10) Anomaly Detection**



By determining the MAE for each sequence and then examining the distribution of MAE values, the code evaluates how well an LSTM model performed on the test data set. It also checks for anomalies based on a predefined threshold for the number of outliers.

**References:**

<https://julien-vitay.net/lecturenotes-neurocomputing/3-deeplearning/6-Autoencoders.html>

<https://en.wikipedia.org/wiki/Long_short-term_memory>

Fig-1 is taken from

<https://lilianweng.github.io/posts/2018-08-12-vae/>