**INFO – 6147**
**Deep Learning with Pytorch**
**Garvit Sakhuja**

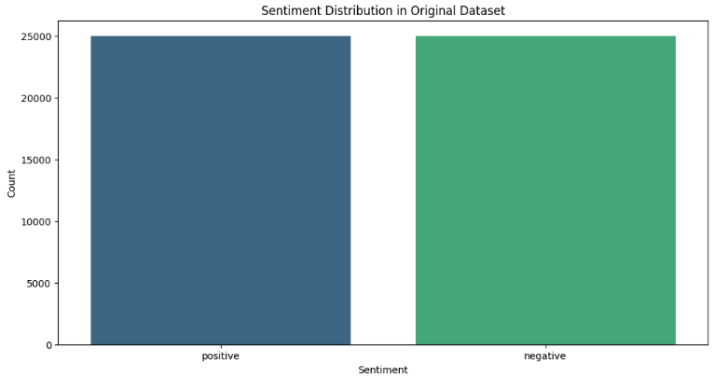
# Sentiment Analysis using Pytorch

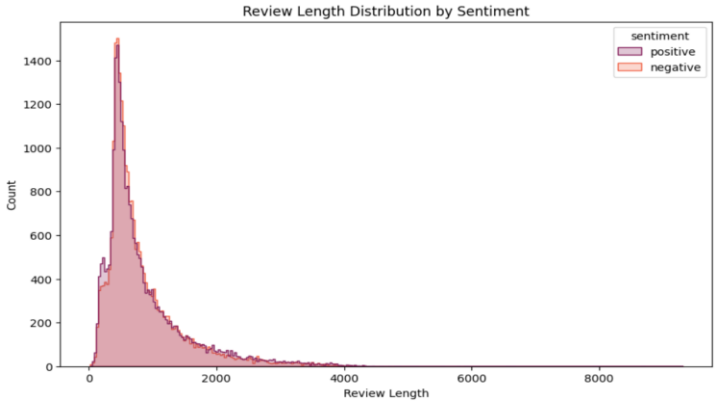
## Introduction

Sentiment analysis is a key task in Natural Language Processing (NLP) that involves determining the sentiment expressed in text data. This project focuses on performing sentiment analysis on IMDb movie reviews using deep learning techniques, specifically Long Short-Term Memory (LSTM) networks implemented in PyTorch. The primary objective is to classify movie reviews as either positive or negative. This classification task is highly valuable for finding insights into public opinion, which can be beneficial for both consumers and industry.
The goal is to develop a deep learning model that is capable of accurately classifying IMDB movie reviews based on the sentiments.

## Dataset

The dataset used for this project is IMDB movie reviews dataset, which contains 50K labels (25K are positive, 25K are negative). The dataset contains two columns 'reviews' (text of the review) and 'sentiment' (positive or negative).



**Data Preprocessing**
For preprocessing of textual data, I took the following steps:

- Lowercasing: All text was converted to lowercase for uniformity.
- Removing Punctuation: Punctuation removed from the text.
- Removing Numbers: Digits and special characters removed.
- Stopword Removal: Common words that do not contribute to sentiment, such as "the" and "is," were removed using NLTK's stopword list.
- Tokenization: Sentences were split into words using a tokenizer from TorchText.
- Padding: Sequences were padded to a uniform length using a special <pad> token to ensure input dimensions were consistent.

**Data Augmentation**
To enhance model performance, text augmentation techniques were applied using nlpaug.

- Synonym Replacement: Replacing words with their synonyms from WordNet.
- Random Deletion: Randomly removing words from the text.

**Model Architecture**
The model architecture is based on an LSTM network implemented in PyTorch. The architecture includes:

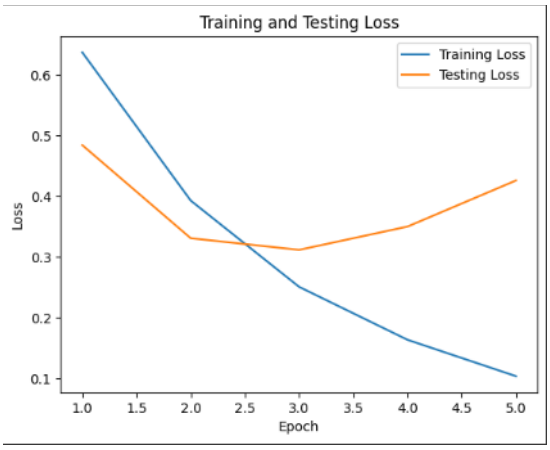
- Embedding Layer: Coverting words to vectors, capturing semantic meanings.
- LSTM Layer: Processing sequential data and capturing dependencies between words and review.
- Batch Normalization: Applied after the LSTM layer to stabilize learning and improve convergence.
- Fully Connected Layer: Maps LSTM outputs to binary sentiment classes (positive/negative).
- Sigmoid Activation Function: Used for binary classification to output probabilities between 0 and 1.

**Model Training**
The model was trained using Pytorch with CUDA support for faster computation on GPUs.

- DataLoader was used to batch input data during training.
- Binary Cross Entropy Loss was used as a loss function for binary classification tasks.
- Adam optimizer with a learning rate of 0.001 was used to update model weights.
- For regularization, weight decay was applied for L2 Regularization to prevent overfitting.
- 5 epochs for training process, with data split of Training (80%) and Testing (20%).

## Model Evaluation

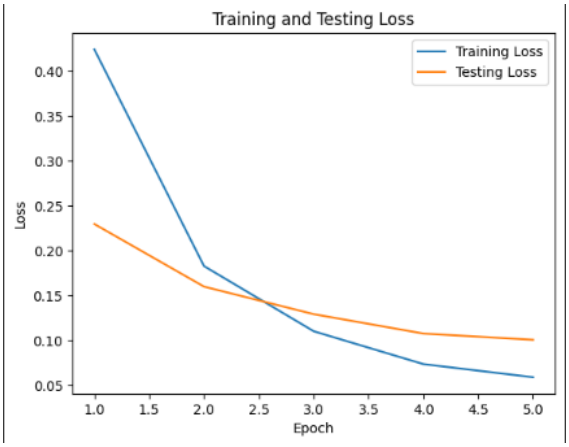
### Original Dataset




- The training loss decreases steadily across epochs but testing loss initially decreases but then starts to increase after third epoch suggesting potential overfitting.
- Performance metrics

- Accuracy: 86
- Precision: 88%
- Recall: 84%
- F1-Score: 86%

### Augmented Dataset




- Both training and testing losses decrease steadily across epochs, indicating improved generalization.
- Performance metrics

- Accuracy: 97%
- Precision: 97%
- Recall: 96%
- F1-Score: 97%

## Conclusion

We were successfully able to create a sentiment analysis model using LSTM networks for IMDB reviews. The model was able to achieve an accuracy of 97% by employing techniques such as data augmentation, regularization, batch normalization, etc. Utilizing techniques such as synonym replacement and random deletion significantly improved model performance by providing more diverse and reliable training data which not only improved accuracy but reduced overfitting due to better generalization.