Emotion Detection of text using Deep Learning

Presented By:

Patheti Satya Sree
R Sreelekshmi
Pedapolu Padma

Agenda

- Introduction
- Problems Addressed
- Literature Survey
- Proposed Approach
- System Workflow
- ML Concepts Involved
- Dataset Description
- Implementation Overview
- Results & Analysis
- Conclusion and Future Work

Introduction

- What It Is?

  Automatic detection of emotions (joy, sadness, anger, fear, disgust, surprise) from text

- Why It Matters?
  - Enhances human–AI interaction
  - Useful in social media analytics, chatbots, customer feedback, mental health applications

- Project Goal:
  - Can we design a system that is both accurate and efficient

Problems Addressed

Challenge	Limitation in Existing Work	Our Focus
Context Understanding	Earlier models lost emotional reversals (“I thought it would be fun, but it wasn’t”)	Evaluate which bidirectional model captures context best
Heavy Architectures	Deep learning models are accurate but slow and parameter-heavy	Lighter models that still maintain strong performance
Unfair Comparisons	Prior studies used different datasets / pipelines	Controlled experiment – same data & setup, only model differs

Literature Survey

- **Evolution of Emotion Detection**

1. Lexicon-Based Approaches (2000s)

- Used emotion dictionaries like *WordNet-Affect* and *NRC Lexicon*.
- Worked well for basic word–emotion mapping.
- ✕ Failed to understand context and negations.
  (Example: "I am not happy" still marked as joy.)

2. Machine Learning Approaches (2010s)

- Algorithms like **SVM**, **Naïve Bayes**, and **Random Forest** used features such as TF-IDF and n-grams.
- Improved over lexicon-based methods but still treated text as “bag of words.”
- ✕ Lost sequence information and long-term dependencies.

3. Deep Learning Era (2016–2020)

- Introduction of **LSTM**, **CNN**, and **Bi-LSTM** models.
- Models could learn context and word order automatically.
- Bi-LSTM, in particular, captured both forward and backward dependencies

4. Bi-LSTM with Emotion-Sensitive Preprocessing (IEEE 2021)

- Combined bidirectional LSTM with advanced text cleaning.
- Achieved around **82% accuracy** on emotion datasets.
- ☑ Showed importance of preprocessing.
- ✕ Did not explore simpler, faster alternatives like Bi-GRU.

5. GRU / Bi-GRU and Hybrid Models (2022–2024)

- Studies tested GRU or GRU–LSTM hybrids for efficiency.
- Found faster convergence and reduced training time.
- ✕ However, they used different datasets and preprocessing, making direct comparison difficult.

Our Contribution

- Addressed the lack of fair comparison between Bi-LSTM and Bi-GRU in text-based emotion detection.
- Implemented both models under identical conditions – same dataset, embeddings, and preprocessing.
- Conducted comparative analysis on accuracy, Precision, Recall and F1-score
- Found that Bi-GRU achieves similar or slightly better accuracy with lower computational cost.

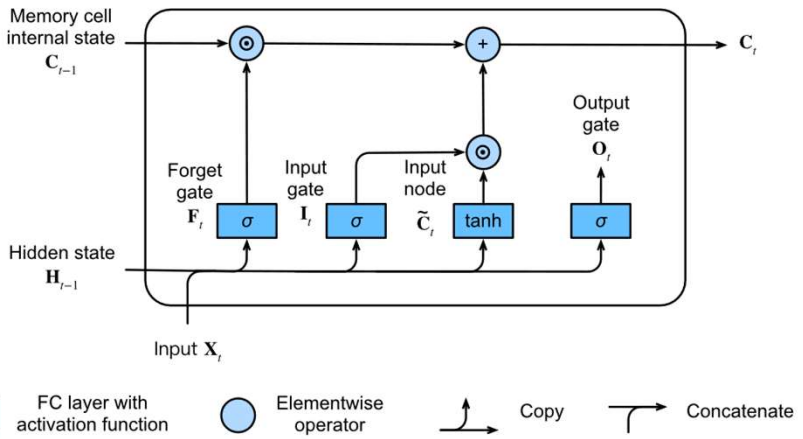
Proposed Methodology

Proposed Methodology

- Collect and preprocess labeled emotion-text dataset.
- Apply extensive text cleaning:
  - HTML tag removal
  - LowercasingStopword removal
  - Stemming
  - Tokenization & padding
  - Train multiple deep-learning architectures:
  - LSTM
  - GRU
  - Bidirectional LST
  - Bidirectional GRU
  - Stacked LSTM/GRU
- Evaluate metrics: Accuracy, Precision, Recall, F1-score.
- Deploy best-performing model (Bi-GRU) for prediction

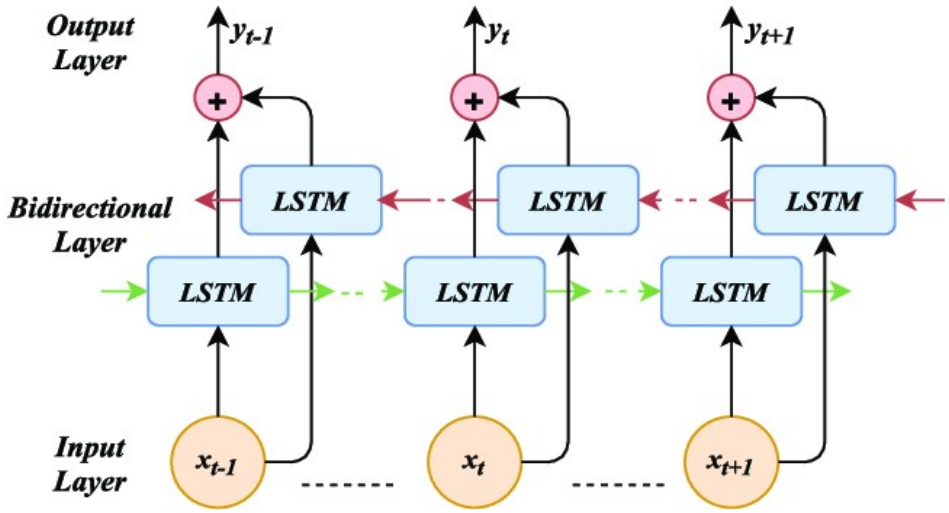
How an LSTM Cell Works

- **Input:** Takes the current word embedding (xtx_txt) and previous hidden state (ht−1h_{t-1}ht−1).
- **Forget Gate (σ):** Decides what old information to discard from memory.
- **Input Gate (σ) & Candidate (tanh):** Adds new important information to the cell state.
- **Cell State Update:** Combines what to keep and what to add → maintains emotional context.
- **Output Gate (σ):** Controls what information to send as the new hidden state (hth_tht).



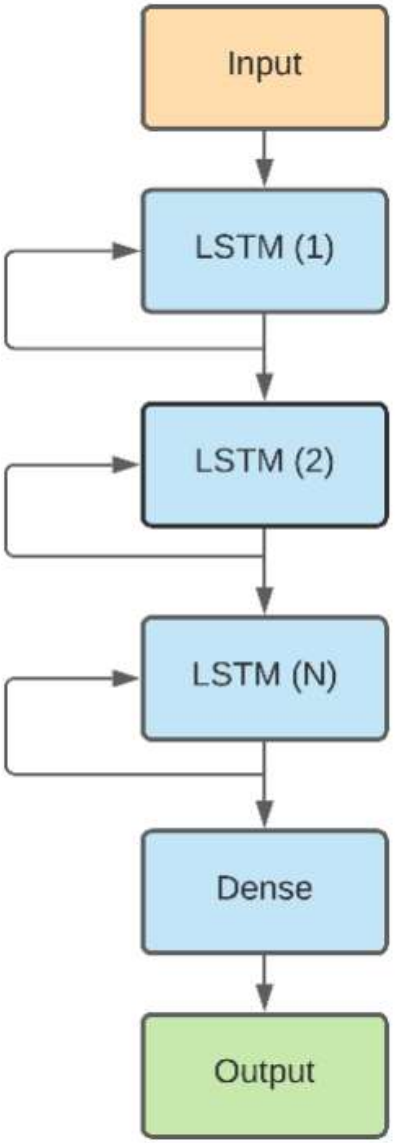
How Bi-LSTM Works

- Bi-LSTM has **two LSTM layers**:
- one processes the sequence **forward** (past → future)
- the other processes it **backward** (future → past).
- Outputs from both directions are **concatenated** and sent to the output layer.
- Captures **complete context** — both previous and next words.



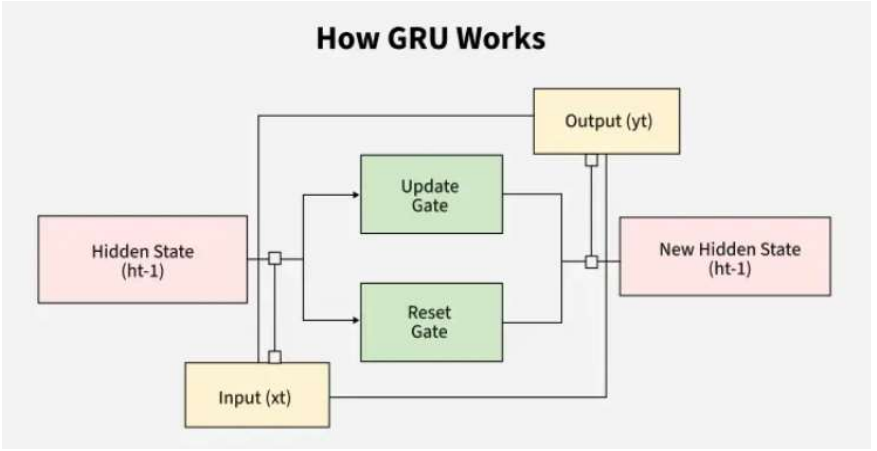
How Stacked LSTM Works

- A **Stacked LSTM** contains **multiple LSTM layers** placed sequentially.
- The **output sequence** from each LSTM layer is **fed as input** to the next layer.
- The **lower layers** capture basic language patterns and word dependencies.
- The **upper layers** learn **higher-level emotional representations** (like tone, intensity, or sarcasm).
- Finally, a **Dense + Softmax layer** predicts the emotion category.



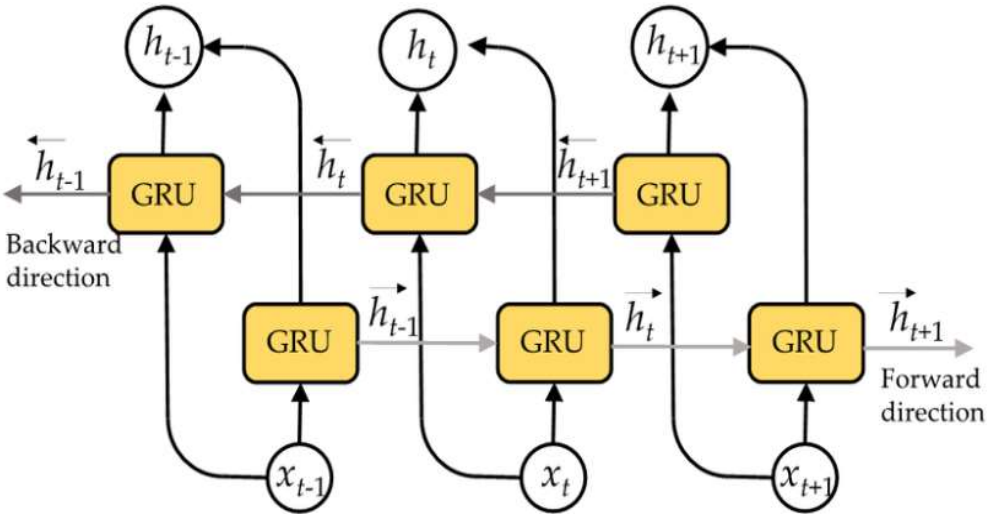
How GRU Works

- GRU (Gated Recurrent Unit) is a **simplified version of LSTM**, designed to solve the **vanishing gradient problem** while being computationally efficient.
- It has **only two gates**:
- **Update Gate:** Decides how much of the past information to keep.
- **Reset Gate:** Decides how much of the previous memory to forget.
- Unlike LSTM, GRU **merges the cell state and hidden state** into one.
- It learns **long-term dependencies** with fewer parameters, making it faster to train.

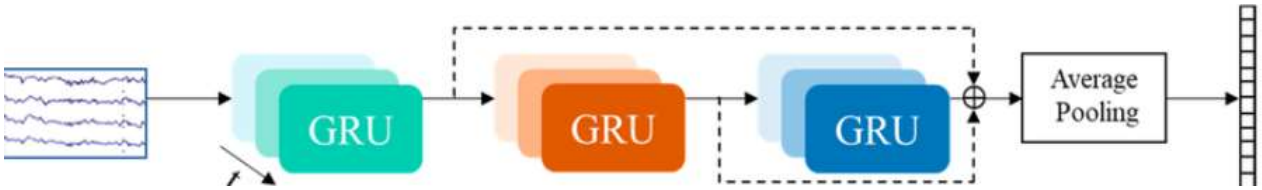


What is Bidirectional GRU?

- A **Bidirectional GRU (Bi-GRU)** runs two GRUs in parallel:
- One reads the input **forward** (left → right).
- The other reads the input **backward** (right → left).
- The outputs from both directions are **combined**, allowing the model to understand **context from past and future words** at every point in the sequence.
- This is especially useful in **emotion detection**, where meaning often depends on both previous and next words.



What is Stacked GRU?

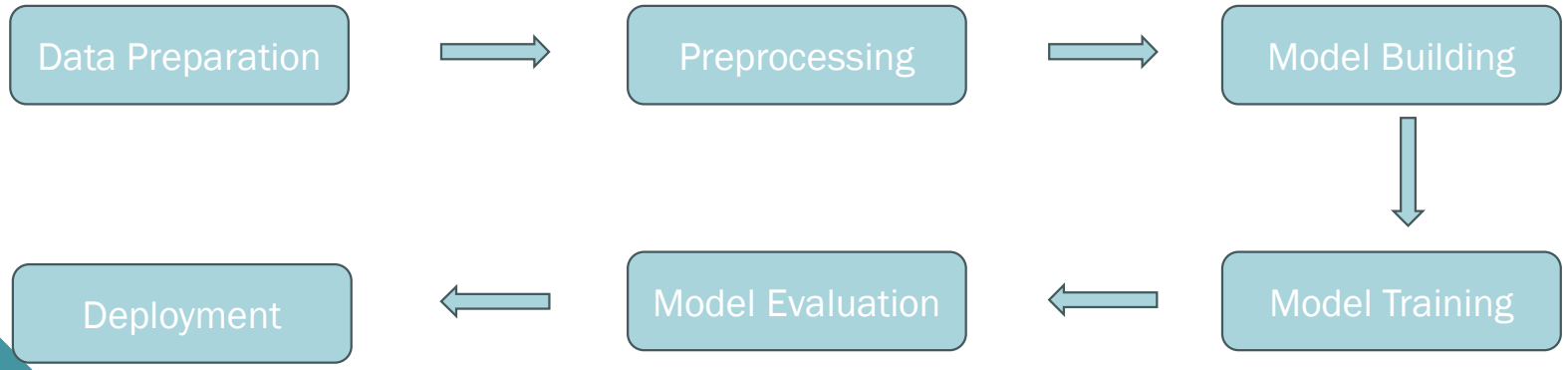


- A **Stacked GRU** (also called **Deep GRU**) is formed by **placing multiple GRU layers on top of each other**.
- The **output sequence** of one layer acts as the **input** to the next layer.
- Lower layers capture **basic temporal patterns**, while higher layers learn **more abstract and complex representations**.
- This hierarchical learning helps in tasks like **emotion classification**, **speech recognition**, and **language modeling**.

Machine Learning Concepts Involved

- Natural Language Processing (NLP)
- Tokenization
- Stopword removal
- Stemming
- Word embeddings
- Sequence Modeling
- LSTM & GRU units
- Bidirectional RNNs
- Stacked recurrent layers
- Deep Learning Concepts
- Embedding layer, Dense softmax classifier
- Categorical cross-entropy loss
- Evaluation Metrics (Accuracy, Precision, Recall, F1-score (macro/micro))

Workflow


Data Preparation
Preprocessing
Model Building
Model Training
Model Evaluation
Deployment

Dataset Description

- Twitter dataset
- six emotions:

  sadness, joy, love, anger, fear, surprise

text	label
i didnt feel humiliated	0
i can go from feeling so hopeless to so damned...	0
im grabbing a minute to post i feel greedy wrong	3
i am ever feeling nostalgic about the fireplac...	2
i am feeling grouchy	3

Fig: Train data screenshot

- Train dataset : (16000, 2)
- Test datset : (2000, 2)
- Val datset : (2000, 2)

Libraries used

- Programming Environment: Colab Notebook

  Libraries used:

  - NumPy, Pandas
  - NLTK for preprocessing
  - TensorFlow Keras for model building
  - Matplotlib

Data preparation and preprocessing

Data preprocessing:

- HTML tags,lowercase,punctuations,tokenization,stop words,stemming
- Rejoined into single string after preprocessing

Tokenization for Neural networks

- Unique index for each words
- Padding

Label encoding

Model Building and Training

All models start with an **Embedding layer**, which converts word indices into dense vector representations.

Models implemented:

- LSTM, GRU
- Bidirectional LSTM/GRU
- Stacked LSTM/GRU

All models are compiled with:

- loss='categorical_crossentropy
- optimizer='adam'
- metrics=['accuracy']

Model Building and Training

Training

- Validation is done.
- Training parameters:
    - epochs = 5
    - batch_size = 32

Model Evaluation

	Model Name	Accuracy Score	F1 Score(macro)	Recall Score(macro)	Precision Score(macro)	F1 Score(micro)	Recall Score(micro)	Precision Score(micro)
0	Bidirectional LSTM	0.8670	0.813646	0.825409	0.805263	0.8670	0.8670	0.8670
1	LSTM	0.4965	0.322173	0.353344	0.342629	0.4965	0.4965	0.4965
2	Stack LSTM	0.3475	0.085962	0.166667	0.057917	0.3475	0.3475	0.3475
3	Bidirectional GRU	0.8890	0.831810	0.810637	0.863913	0.8890	0.8890	0.8890
4	GRU	0.3475	0.085962	0.166667	0.057917	0.3475	0.3475	0.3475
5	Stack GRU	0.3475	0.085962	0.166667	0.057917	0.3475	0.3475	0.3475

Fig: Results of all the models

Deployment

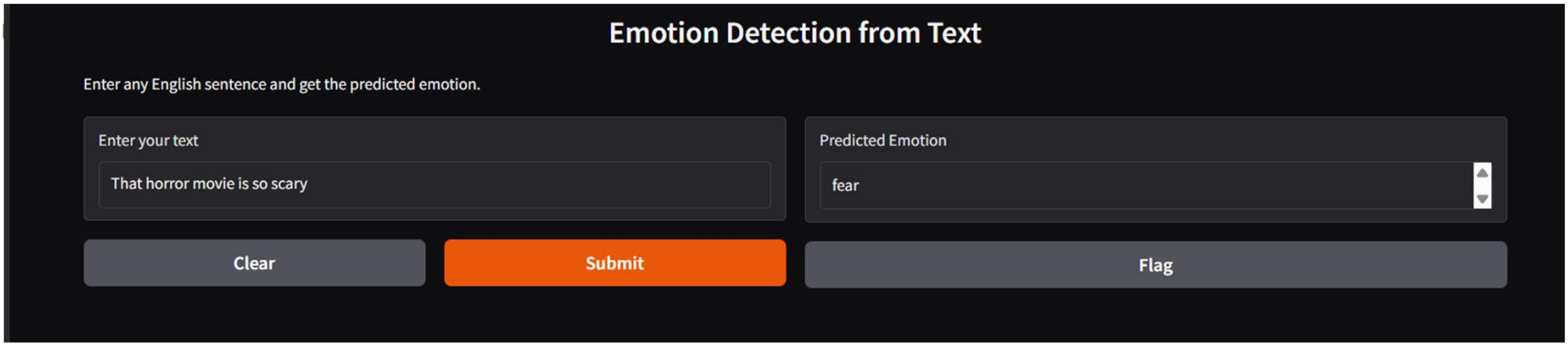


Fig: Deployment on Gradio

Conclusion

- Successfully built an emotion classification system using deep learning.
- Bidirectional GRU achieved the best performance among all tested models.
- System can predict 6 emotions reliably from input text.

Future Work

- Use pre-trained embeddings like GloVe or BERT for improved accuracy.
- Implement attention mechanisms for better context capture.
- Expand dataset with multilingual support.
- Deploy as a web app or integrate into chatbots.
- Try transformer-based architectures for state-of-the-art results.

Thank you