Course Project - Abstract

Team Members:

Katta Vibhas Reddy – 1602-23-737-189

Chidurala Thrilochan – 1602-23-737-187

Problem Statement

Create an intelligent system that predicts the next word in a sentence using a Recurrent Neural Network (RNN). The system should accept user input text, process it, and suggest the most probable next word(s) based on trained language models

---

Hardware Requirements

Component	Specification
Processor	Intel i5 or higher / AMD equivalent
RAM	Minimum 8 GB
Storage	Minimum 5 GB free space
GPU (Optional)	NVIDIA GPU for faster model training

Software Requirements

Component	Specification
OS	Windows 10 / Ubuntu 20.04 / MacOS
Python Version	Python 3.8 or above

Libraries / Frameworks	TensorFlow / PyTorch, Keras, NumPy, pandas, scikit-learn, NLTK / spaCy, matplotlib
IDE	Jupyter Notebook / VS Code / PyCharm
Web Framework (optional)	Flask / Streamlit (for UI deployment)

System Requirements

Accept partial text input from the user.

Preprocess the input (tokenization, lowercasing, padding, etc.).

Use an RNN (e.g., LSTM / GRU) model trained on a text corpus.

Predict the most probable next word (or top N suggestions).

Display the prediction(s) to the user with option to continue typing.

Algorithm

1. Text Input & Preprocessing

Input: A sentence fragment typed by the user.

Steps:

Tokenize the text into words.

Convert words into integer indices using a vocabulary (Tokenizer).

Pad/truncate sequences to a fixed length for the model.

---

2. Model Training (RNN-based Language Model)

Use a dataset (e.g., Wikipedia corpus, movie subtitles, or custom dataset).

Train an RNN / LSTM / GRU model to learn word sequences.

Objective: Minimize categorical cross-entropy loss for next-word prediction.

---

3. Next Word Prediction

Pass the preprocessed user input through the trained model.

Output the probability distribution over the vocabulary.

Select the top predicted word(s) using:

Argmax (single best word), or

Beam Search / Top-K Sampling (multiple suggestions).

---

4. Output Generation

Display the predicted word to the user.

Optionally allow autocomplete (append word to sentence).

Enable iterative continuation of text with repeated predictions.

Example Mapping

For input:

The weather is

Predicted outputs:

"nice"

"bad"

"sunny"

---

5. Output

Show next-word suggestion(s) in UI.

Provide option to continue generating text.

Allow saving the generated sentence to a file (optional).