# Lab Exercise 1 - Train an RNN Model for Text Classification

### Goal

Train your own deep-learning model (RNN) on a text dataset and test its accuracy.

### What you will do

1. **Load a dataset**
   Example: IMDB movie reviews (Positive/Negative).
2. **Split the dataset**
   - 80% Training
   - 20% Testing
3. **Preprocess the text**
   - Convert text to sequences of numbers
   - Pad all sequences to the same length
4. **Build the RNN model**
   - Embedding layer
   - LSTM or GRU layer
   - Final output layer
5. **Train the model**
   - Use the training set
   - Plot loss and accuracy
6. **Test the model**
   - Use the test set
   - Calculate accuracy

# Lab Exercise 2 - Run LLAMA Locally Using Ollama & Compare With RNN

### Goal

Run a pre-trained LLaMA model on your own laptop using **Ollama**, and compare its performance with the RNN you trained in Lab 1.

### What you will do

### Step 1 - Install Ollama

Download from: https://ollama.com/download

Run:

ollama run llama3

**Step 2 - Test the LLaMA model on your dataset**

For each test example:

Send the text to LLaMA with a prompt like:

Classify the sentiment of this review as Positive or Negative:
<review>
Answer with only one word.

Record:

- LLaMA's prediction
- Whether it matches the true label

**Step 3 - Calculate accuracy**

Just like the RNN model, compute:

Accuracy = correct_predictions / total_test_samples

**Step 4 - Compare RNN vs LLaMA**

Create a small table:

| Model | Accuracy | Strengths | Weaknesses |
|---|---|---|---|
| RNN (your model) | ? | simple, low cost | lower accuracy |
| LLaMA (pretrained) | ? | high accuracy, understands text better | needs GPU/CPU |

# Lab Exercise 3 - Build a Simple RAG System (Retrieval + LLaMA)

**Goal**

Build a Question-Answer system that uses your documents + LLaMA.

**What you will do**

**Step 1 - Prepare your documents**

Collect 3–5 text files (notes, PDFs converted to text).

Break them into **small chunks** (e.g., 200–300 words).

**Step 2 - Create embeddings**

Use SentenceTransformers:

```
from sentence_transformers import SentenceTransformer

model = SentenceTransformer("all-MiniLM-L6-v2")
embeddings = model.encode(chunks, convert_to_numpy=True)
```

**Step 3 - Build a simple similarity search**

For a user question:

- Convert question into embedding
- Compare with chunk embeddings
- Retrieve top 3 relevant chunks

**Step 4 - Build the RAG prompt**

Send this to LLaMA:

```
Use only the following context to answer the question:

<context from top chunks>

Question: <user question>
Answer:
```

**Step 5 - Test RAG vs No-RAG**

Create 10 questions based on your documents.

Compare:

| Method | Correct? | Notes |
|---|---|---|
| LLaMA alone | ? | May hallucinate |
| LLaMA + RAG | ? | More accurate |