TEXT SUMMARIZATION WITH USING MULTI-NEWS DATASET REPORT

FINAL PROJECT REPORT

2025W-T3 AML 3104 - Neural Networks and Deep Learning 01

(DSMM Group 1)

Presented by:

Jashanpreet Kaur (C0928354)

Ashish Kathrotiya (C0924786)

Devansh Patel (C0928483)

Govind Dogra (C0929421)

Shlok Shah (C0925532)

Submitted to:

Ishant Gupta

1. Introduction

The Text Summarisation Model developed with Keras/TensorFlow is explained in detail in this document. The model uses a sequence-to-sequence (Seq2Seq) architecture with LSTM-based encoder-decoder components to produce succinct summaries from news items.

Objective

- **Input**: A news article (text).
- **Output**: A short, coherent summary.

Dataset

- **CNN/DailyMail Dataset** (from Kaggle) containing news articles and their corresponding summaries.

2. Data Preprocessing

Data Loading & Filtering

- Pandas is used to load the dataset.
- Overlength articles and summaries (TEXT_LIMIT=1600, SUMMARY_LIMIT=500) are eliminated.
- Seaborn and Matplotlib are used to visualise length distributions.

Train-Test Split

- The dataset is divided into:
    - The majority of the dataset is used for training.
    - The last ten samples are the test data (for validation).

Preparing the Sequence and Tokenisation

- Text is transformed into sequences using the Keras tokeniser.
- Summaries now include special tokens (<start>, <end>, <PAD>.)
- To guarantee consistent sequence lengths, padding is used.

Lengths of Sequences and Vocabulary

- The encoder tokeniser is used to determine the size of the input vocabulary.
- The decoder tokeniser was used to determine the output vocabulary size.

- **Maximum Lengths of Sequences:**
    - Max_input_seq_length is the input (Article).
    - Max_output_seq_length is the output (summary).

3. Model Architecture

Encoder-Decoder with Attention

The model consists of:

Encoder (LSTM-based):

- Takes an input sequence (article).
- Produces hidden states (h, c) and encoder outputs.

Decoder (LSTM-based):

- Takes the encoder's hidden states and generates summaries.
- Uses Bahdanau Attention to focus on relevant parts of the input.

Attention Mechanism:

- Computes a context vector to weigh encoder outputs dynamically.

Key Components

Components	Description
Encoder	Embedding + LSTM
Decoder	Embedding + LSTM + Dense (Softmax)
Attention	Dense layers + Softmax weighting

Training Process

- **Loss Function:** Sparse Categorical Crossentropy (since outputs are sequences).
- **Optimizer:** Adam.
- **Batch Size:** 32.
- **Epochs:** 5 per training cycle (total 25 epochs).

4. Model Performance

Training Metrics

Epoch	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
1-5	**~59% → 66%**	**~62% → 66%**	**~2.38e-5**	**~2.98 → 2.53**
6-10	**~67% → 71%**	**~62% → 66%**	**~2.38e-5**	**~2.52 → 2.57**
11-15	**~73% → 87%**	**~62% → 66%**	**~2.38e-5**	**~2.60 → 3.20**

Observations

- **Training Accuracy** improves significantly (59% → 87%).
- **Validation Accuracy** plateaus at ~66%, indicating overfitting.
- **Loss** increases on validation data, suggesting the model struggles to generalize.

5. Inference & Testing

Summary Generation

Summaries are produced by the model using:

- At each stage, Greedy Search chooses the word with the highest likelihood.
- When the token is anticipated, it stops.

Example outputs

Example 1 (Index 1116):

Input Text:
"Rescue workers have pulled a body from underneath the rubble of a collapsed apartment building in Cologne, Germany..."

Generated Summary:
"New the death toll rises to 931 according to Bangladesh's national weather service..."
❌ Incorrect facts (hallucination).

Reference Summary:
"Rescue workers pull body from rubble of collapsed building. One person still missing..."

Example 2 (Index 100):

Input Text:
"Michael Jackson's brothers are working on a reunion tour to perform their old Jackson 5 songs..."

Generated Summary:
"Timothy Tracy was accused of fomenting unrest in Venezuela..."
❌ Completely unrelated summary.

Reference Summary:
"Jermaine Jackson says the brothers are meeting Monday to plan a new tour..."

6. Issues & Improvements

Identified problems

- **Overfitting:** High training accuracy but stagnant validation accuracy is known as overfitting.
- **Hallucinations:** The model produces inaccurate or irrelevant summaries.
- **Limited Generalisation:** Faces challenges while dealing with unknown data.

Possible Solutions

Problem	Solution
Overfitting	Add Dropout, Regularization, or Early Stopping.
Hallucinations	Use Beam Search instead of greedy decoding.
Generalization	Increase dataset size or use pre-trained embeddings
Training Instability	Adjust learning rate or use gradient clipping.

7. Web Application

Objective

The purpose of this app is to generate concise summaries from longer news articles using a trained sequence-to-sequence model with attention. The app is built using Streamlit, making it interactive and user-friendly.

Key Functionalities

Model and Tokenizer Loading:

- The function load_model_and_tokenizers() is used to load the trained Keras model (Teacher_Model.keras) and the associated encoder_tokenizer and decoder_tokenizer from pickle files.
- It uses @st.cache_resource to ensure that the model and tokenizers are loaded only once, improving performance.

Summary Generation Logic:

- The function generate_summary(text) takes in an article as input and performs the following:

    - Tokenizes and pads the input article.

- Passes it through the encoder to get hidden and cell states.
- Uses these states to initialize the decoder.
- Iteratively generates tokens one by one until the end token is predicted or the maximum length is reached.
- Decoded token IDs are converted back to words using index_to_word mapping to produce the final summary.

User Interface:

- The Streamlit app includes a simple interface:
  - A text area for users to paste or type the news article.
  - A button to trigger summary generation.
  - A loading spinner for user feedback while the summary is being generated.
  - A result section displaying the generated summary once it's ready.
- It uses st.set_page_config() to configure the layout and title of the app.

Technologies Used

- **Streamlit:** For building the web interface.
- **TensorFlow / Keras:** For the deep learning model used in summarization.
- **Pickle:** To load pre-trained tokenizers.
- **Numpy:** For array manipulation.

8. Conclusion

- Despite overfitting and hallucinations, the model is able to learn to produce summaries.
- **Upcoming enhancements:**
  - Improved attentional systems (based on transformers).
  - larger dataset or improving data specialised to a given domain.
  - post-processing (optimisation of ROUGE scores).
- **Next Steps:**
  - Hyperparameter tuning (LSTM units, embedding size)
  - Try out different Transformer models, like as T5 and BART.
  - Implement as a real-time summarisation API.