Advanced Data Analysis And Machine Learning
Cancer Document Classification

November 18, 2024

Bright Wiredu Nuakoh

In this task, our goal is to classify biomedical text documents, specifically abstracts and full papers that are six pages or fewer. We aim to identify the type of cancer discussed in these papers—Thyroid Cancer, Colon Cancer, or Lung Cancer—using Long Short-Term Memory networks (LSTMs) and Recurrent Neural Networks (RNNs) to capture the sequential dependencies between characters.

Data Description And Visualization

The dataset provides rich textual information to train models for predicting the specific class based on the content of biomedical publications. Figure 1 shows the frequency of publications across three cancer types: Colon Cancer, Lung Cancer, and Thyroid Cancer. The Thyroid Cancer class has the highest number of publications with a value of 2810, followed by Colon Cancer with 2580 entries. Lung Cancer has the lowest frequency, with 2180 publications. The chart highlights slight imbalance in the dataset, as the number of entries varies across the three classes, however this imbalance is not severe so this would not affect the models that bad.

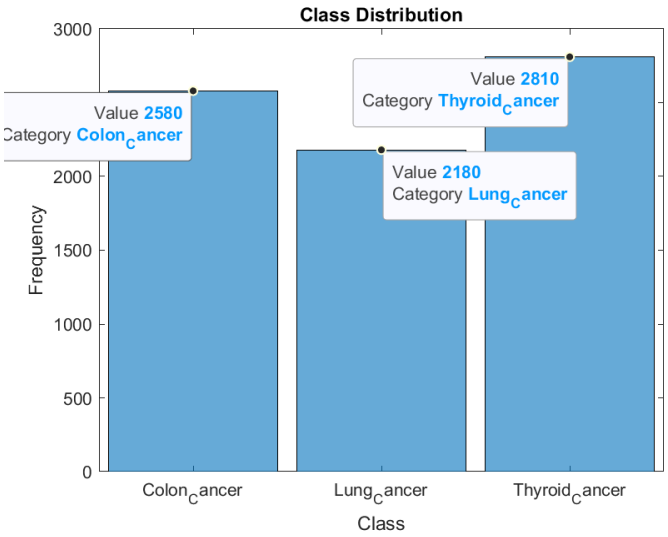

Figure 1: Distribution of classes in our data

From Figure 2, the word clouds illustrate key research themes in colon, lung, and thyroid cancers. Common focus areas include cellular mechanisms, tumor biology, gene expression, and patient outcomes, while each cancer type emphasizes specific aspects, such as genetic mutations in lung cancer and protein activity in thyroid cancer. Overall, the research highlights shared molecular and cellular priorities across these cancers.

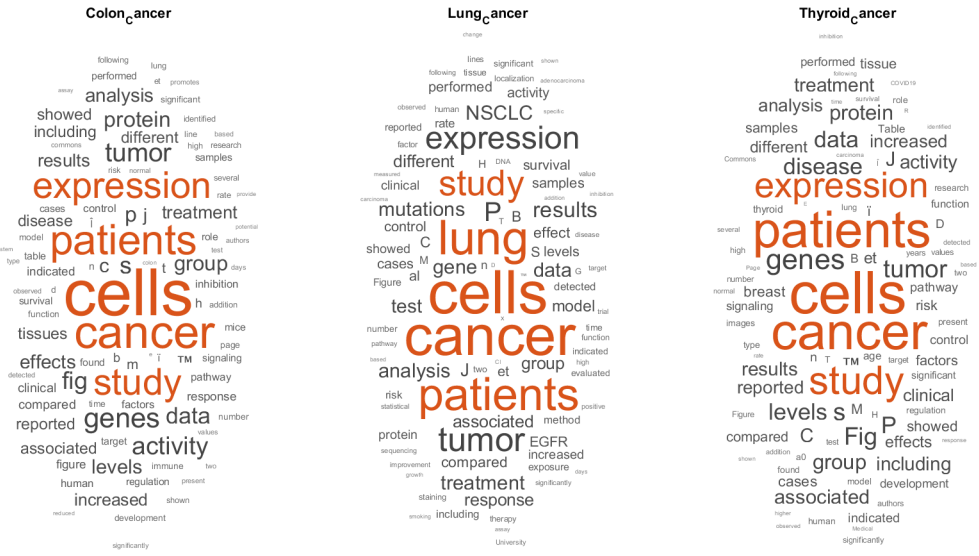

Figure 2: Word cloud visualization of raw data against cleaned data (cleared vocabulary)

Data Preprocessing

We implemented two seperate data preprocessing to suit the LSTM model and the RNN model seperately. The goal was to make the data feasible for each model.

The two functions preprocess text data for different deep learning models. The rnnpt function is tailored for RNN training: it tokenizes the input, removes stop words, filters out short words (less than 4 characters) and long words (more than 8 characters), converts text to lowercase, and removes punctuation. This cleaning helps focus on meaningful features. The preprocessText function is simpler, tokenizing text, converting to lowercase, and erasing punctuation, making it more suitable for general tasks like LSTM training, where capturing longer dependencies might be more important.

Most of the training documents contain fewer than 156 tokens, making 156 a suitable choice for the sequence length, as shown in Figure 3. By setting the sequence length to 156, we standardize the input size for all documents, ensuring that shorter documents are padded to this length ensuring that RNN handle input data uniformly.

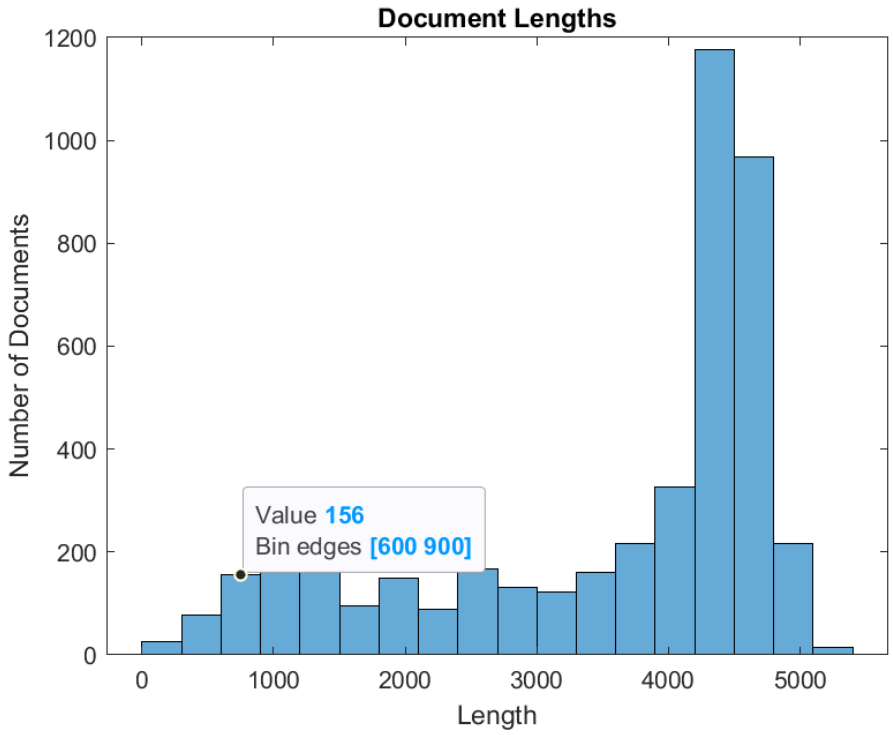

Figure 3: Histogram of lengths for Vacubularies in Processed Data

Model Definition And Strategy

Our modeling goal is implementing a deep learning architecture for text classification using Long Short-Term Memory (LSTM) networks. The model consists of several key layers: a sequence input layer to accept tokenized text data, a word embedding layer to convert words into dense vector representations, an LSTM layer to capture sequential dependencies in the text, followed by a fully connected layer for classification, a softmax layer to output class probabilities, and a classification layer for final decision-making. The network is trained using the Adam optimizer with a mini-batch size of 60, gradient clipping to prevent exploding gradients, and a maximum of 5 epochs. The architecture was designed for efficient processing of text data, leveraging the LSTM's ability to model long-range dependencies while optimizing for fast training and classification accuracy. We will implement RNN model to test compare the prediction accuracy of our LSTM model.

Recurrent Neural Network (RNN)

(RNN) is designed to handle sequential data by maintaining a hidden state that gets updated with each time step. The key idea is to introduce a loop in the network that

allows information to persist over time. For a sequence of inputs $x_1, x_2, \ldots, x_T$ at time steps $t = 1, 2, \ldots, T$, the RNN updates its hidden state h_t recursively:

$$h_t = f(W_h h_{t-1} + W_x x_t + b)$$

where h_t hidden state at time step t, h_{t-1} gidden state from the previous time step, W_h Weight matrix for the previous hidden state, W_x weight matrix for the current input, b bias term and f Nonlinear activation function, typically tanh or ReLU

The output at each time step is computed as:

$$y_t = W_y h_t + c$$

where y_t Output at time step t, W_y Output weight matrix and c Bias for the output layer. The RNN's hidden state captures information from the sequence of inputs and feeds this information forward to make predictions at each time step.

Long Short-Term Memory (LSTM)

LSTMs are a type of RNN designed to overcome the vanishing gradient problem and capture long-term dependencies in sequences. An LSTM introduces memory cells to store information over time, and gates (input, forget, and output gates) control the flow of information. At each time step t, an LSTM computes the following:

Forget Gate:

$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f)$$

where f_t is Forget gate output and σ is Sigmoid function.

Input Gate:

$$i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i)$$

where i_t is input gate output

Cell State Update:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

where C_t is memory cell state at time step t, $\tilde{C}_t$ Candidate memory content, often $\tanh(W_c h_{t-1} + U_c x_t + b_c)$.

Output Gate:

$$o_t = \sigma(W_o h_{t-1} + U_o x_t + b_o)$$

where $o_t$is Output gate output

Hidden State Update:

$$h_t = o_t \cdot \tanh(C_t)$$

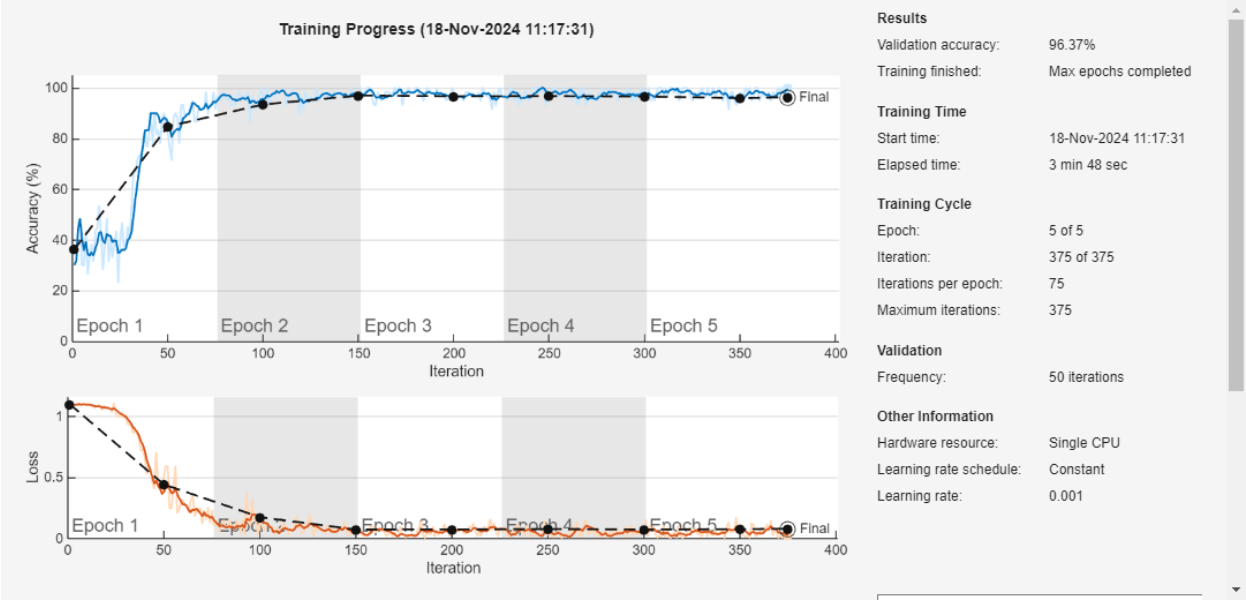

Figure 4: Training details for LSTM model

where h_t is the updated hidden state at time step t

Training The LSTM

From Figure 4 below, the training progress of an LSTM model is shown, achieving a validation accuracy of 96.37% after completing five epochs with a total of 375 iterations. The accuracy plot indicates rapid improvement in the initial epoch, with stabilization occurring by epoch 3, reflecting effective learning. The loss plot displays a steep decline early in training, followed by gradual convergence to minimal values, indicating efficient optimization. The model used a constant learning rate of 0.001 and was trained on a single CPU, completing the process in 3 minutes and 48 seconds. These results demonstrate that the training was both efficient and successful, with the model achieving high accuracy and convergence.

MODEL EVALUATION

The confusion matrix shows the performance of the model on the test set for predicting three classes: Colon Cancer, Lung Cancer, and Thyroid Cancer. The model correctly predicted 496 out of 516 Colon Cancer cases, 414 out of 416 Lung Cancer cases, and 563 out of 582 Thyroid Cancer cases. Misclassifications include 20 Colon Cancer cases predicted as Thyroid Cancer, 2 Lung Cancer cases misclassified as Thyroid Cancer, and 19 Thyroid Cancer cases incorrectly identified as Colon Cancer. The overall performance demonstrates strong classification ability, with minimal misclassification across all classes,

further supporting the model's high validation accuracy of 96.37

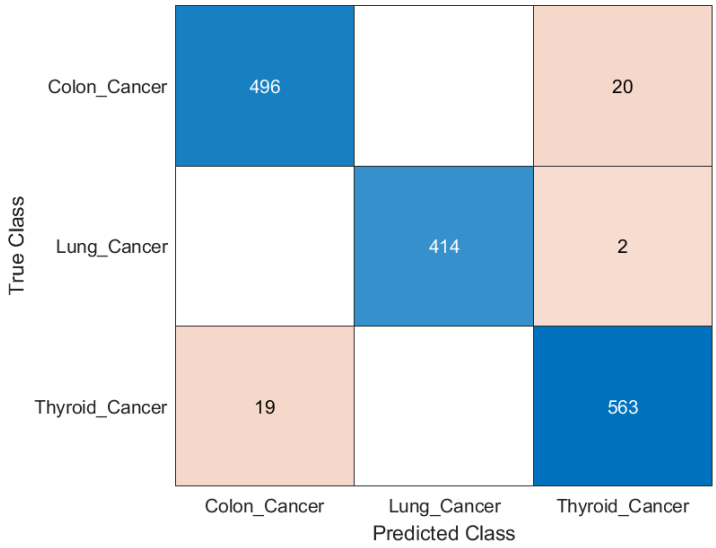

Figure 5: Confusion matrix showing the accuracy for each predicted Label by the LSTM.

The F1 score for each class is calculated using the formula:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Precision}(P) = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

$$\text{Recall}(R) = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

For Colon Cancer the precision was 0.9631, and recall of 0.9612 with an F1 score of 0.9622. For Lung Cancer the precision was 1.0, and recall of 0.9995 with an F1 score of 0.9976. And for Thyroid Cancer the precision was 0.9657, and recall of 0.9674with an F1 score of 0.9665. The high F1 scores across all classes confirm the model's strong performance.