Sentiment Analysis of IMDB Dataset

1. **Preprocessing:**

   IMDB is a dataset of 25000 positive and 25000 negative review. I am performing preprocessing separately for training and testing but the procedure is almost same except tokenization.

   - Collected data from IMDB dataset locally and saved into training and testing data.
   - Removed HTML tags from review.
   - Converted entire review into lowercase.
   - Removed any string punctuation from review.
   - Performed stemming on the review.
   - Applied tokenization on data.
   - Saved tokenization in training part, saved under model directory "tokenizer.pickle" and used the same file in testing.

2. **Design Choice and Architecture for network:**
   - Performed deep learning convolutional model with RNN LSTM model on IMDB data to get the maximum accuracy.
   - RNN model will just not help with building a model that consider the individual words but also remember the order they appear in which leads to a powerful model for sentiment prediction.
   - I have used LSTM model as an RNN model as we shown above it is good tool for anything that has a sequence in it as the meaning of a word depends on the ones that preceded it.

   - **Architecture:**
   - **Embedding layer**: It is used to learn word embedding and perform NLP tasks such as text classification, sentiment analysis, etc.
     - Input dimension: 5000
     - Output dimension: 32
     - Input length: 500
   - **Dropout**: Added 20% drop rate to prevent overfitting
   - **CNN layer**:
     - Filter size: 32
     - Activation: Relu
   - **Maxpooling:** Added with pool size 2 which helps to extract complex features.
   - **LSTM**: Added with 128 units.
   - **Dropout**: Added 20% drop rate to prevent overfitting

- **Output layer:**
    - Added dense layer as output layer with sigmoid activation function.
- **Optimizer**: Adam
- **Loss function**: binary_crossentropy as we are having only two labels positive and negative.

3. **Training and testing accuracy:**
    - Below training accuracy observed with 3 epochs as increasing more epochs leads to more time consumption and comprise in accuracy as well as increasing input dimension and input length in embedding layer result in less accuracy.
    - Training accuracy: 91.57%
    - Testing accuracy: 87.79%