School of Computer Sciences

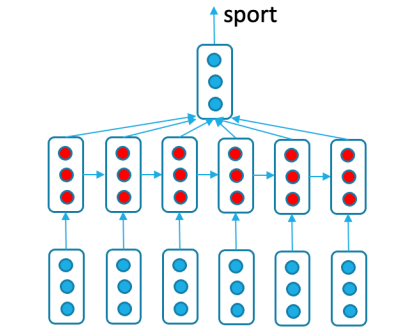
CPC353 Natural Language Processing

Lab: Text and Speech Analytics Semester I 2025/2026

Duration: 1 hour

Objectives

1. Learn to build an LSTM text classifier using word embedding features.



Text classifier

Data

Message	Rating
i love the movies!	Good
The actors are great.	Good
Beautiful actress :)	Good
i do not like the music	Bad
nice story	Good
actors are great, but overall is not nice.	Bad
love it!	Good
great...	Good
enjoy it very much.	Good
Wonderful experience.	Good
really boring	Bad
:(	Bad
Bad acting	Bad
I do not like the actors	Bad
Fall asleep throughout the movie!	Bad
too much dialogs and not much actions	Bad

Steps

1. Open your Python IDE
2. Install tensorflow, genism, nltk, and scikit-learn package
3. Open a new python script lstm_text_classifier.py.
4. Set the following parameters:
   a. sentence_length = 15
   b. n_embedding = 50
   c. n_output = 2
   d. batch_size = 4
5. Import gensim.downloader, and download the pretrained word embedding model "glove-wiki-gigaword-50" by calling downloader.load("glove-wiki-gigaword-50") and assign it to model_glove. You need to have Internet connection to load it for the first time.
6. Import TweetTokenizer from nltk and create an instance of it.
7. Normalize the sentences to lower case.
8. Tokenize the given sentences.
9. Convert the tokens to word embedding vectors by calling model_glove[tokens]. Note that model_glove is the word embedding model and tokens are tokenized sentence.
10. Convert the tokenized sentences to numpy array, X.
11. Pad the numpy array X, so that the length of the array to the maximum length of the sentence, e.g. 15 by calling tensorflow.keras.preprocessing.sequence(X, maxlen=sentence_length)
12. Print the shape of X.
13. Create the following layers:
    a. inputs = Input(shape=(sentence_length, n_embedding))
    b. lstm = LSTM(2, return_sequences=True, return_state=True)
    c. outputs_seq, state_h, state_c = lstm(inputs)
    d. flat = Flatten()(outputs_seq)
    e. outputs = Dense(n_output, activation='softmax')(flat)
    f. model = Model(inputs=inputs, outputs=outputs)
14. Refer to previous lab where we train the feedforward neural network, and apply the same step here. Use ‘adam’ optimizer, set the number of epoch to 20, batch_size=4.
15. Refer to previous lab to do testing.