PRASANNA, LAKSITA
3/20/25

DEEP LEARNING HW 4

In this homework assignment, sentiment classification was implemented on the Twitter US Airline Sentiment data using Recurrent Neural Networks (RNNs). In addition, the difference between training a word embedding layer from scratch and using a pretrained embedding layer (GloVE) was examined.

Problem 1 (Training embedding layer from scratch)

In this problem, the best model from Problem 2 of Hw3 (higher learning rate) was used. Along with that, word embedding layer was trained from scratch instead of using one-hot-encoding.

max_length = 30

epochs = 20

Embedding parameters = 100 Dimension of embedded vectors

Table 1: LSTM with word embeddings

Model	Learning rate	Epochs before overfitting	Train acc.	Val acc.	Test acc.
LSTM with word embeddings	0.001	0.5	0.9498	0.7636	0.7549

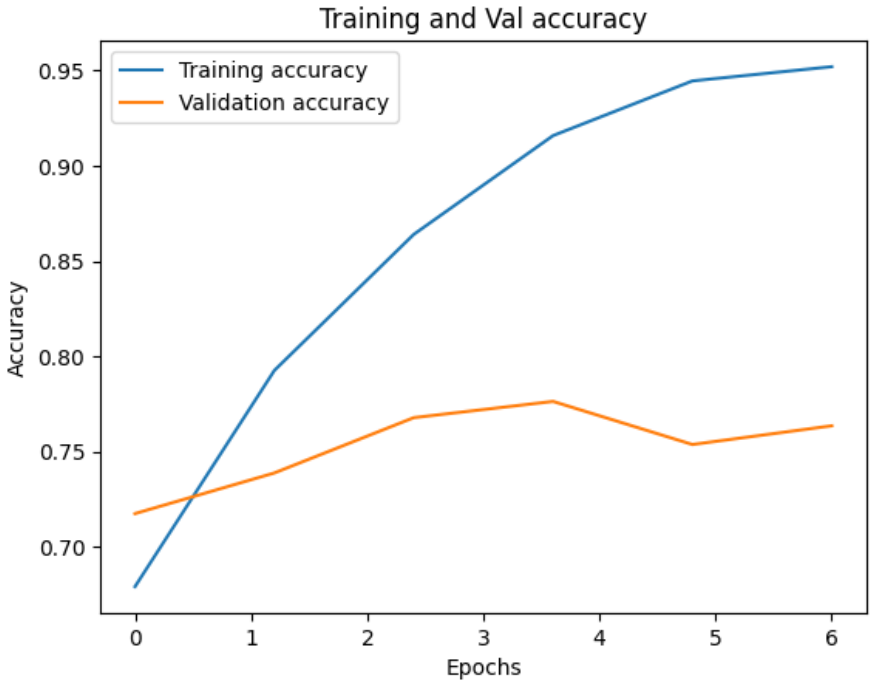

Figure 1: LSTM with word embeddings

Problem 2 (Frozen weights)

In this problem, GloVE was used for the word embeddings as it's a pre-trained model. For this part, the weights were frozen by setting 'trainable '= 'False' as one of the inputs for the Embedding layer.

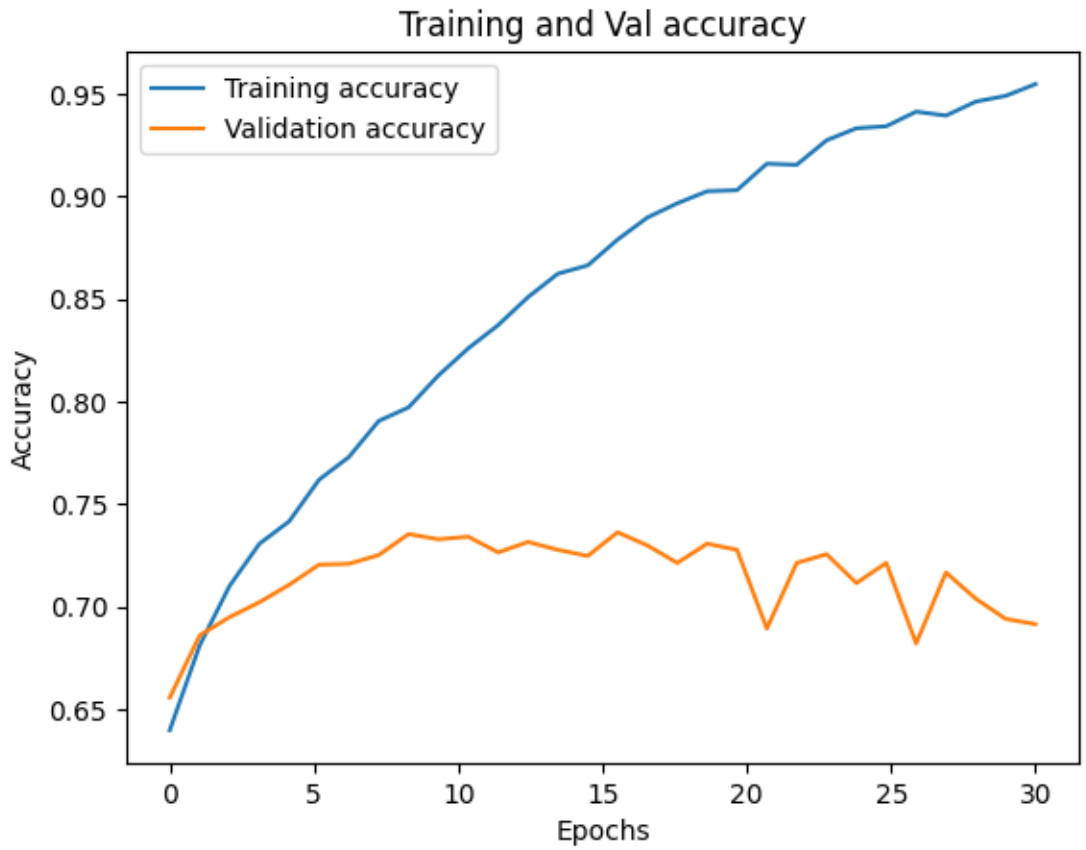

Figure 2a) LSTM with GloVE (frozen weights)

Problem 2 (Fine tune weights)

In this problem, GloVE was used for the word embeddings as it's a pre-trained model. For this part, the weights were fine-tuned by setting 'trainable '= 'True as one of the inputs for the Embedding layer.

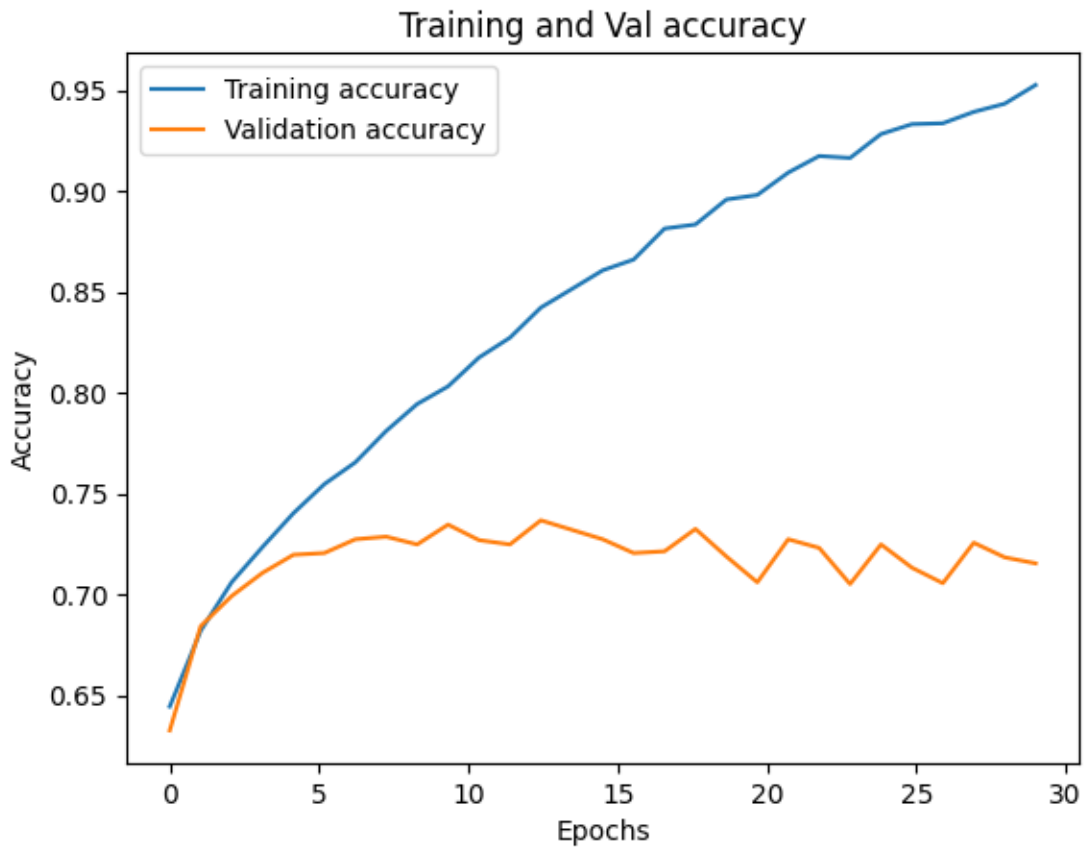

Figure 2b) LSTM with GloVE (fine-tune weights)

Table 2: GloVE word embeddings with LSTM

Model	Learning rate	Epochs before overfitting	Train acc.	Val acc.	Test acc.
LSTM with GloVE (frozen)	0.001	1	0.9558	0.6914	0.7014
LSTM with GloVE (fine-tune)	**0.001**	**1**	**0.9577**	**0.7153**	**0.7308**

When the pretrained GloVE embedding layer was used, fine-tuning the weights proved to have significantly increased Training, Validation, and Test accuracies more than the model with frozen weights. Since the LSTM with GloVE (fine-tune) model had the highest test accuracy, it is more generalizable on unseen data.

```
from tensorflow.keras.optimizers import Adam

#Building the base_model
base_model = Sequential([
  LSTM(32, activation='tanh', input_shape=(max_length, vocab_size), return_sequences=True),
  LSTM(32, activation='tanh', input_shape=(max_length, vocab_size)),
  Dense(256, activation='relu'),
  Dense(3, activation='softmax')])
base_model.compile(loss='categorical_crossentropy', optimizer=Adam(learning_rate=0.001), metrics=['accuracy'])

base_model_callback = myCallback(model=base_model)
callback.append(base_model_callback)

history = base_model.fit(train_generator, epochs=epochs, verbose=1, validation_data=val_generator, callbacks=callback[0])
```

Figure 3) Higher learning rate model network

Table 3: Comparing all models

Model	Learning rate	Epochs before overfitting	Train acc.	Val acc.	Test acc.
Higher LR (Hw 3)	**0.001**	**2.5**	**0.9596**	**0.7644**	**0.7832**
LSTM with word embeddings	0.001	6	0.9498	0.7636	0.7549
LSTM with GloVE (frozen)	0.001	1	0.9558	0.6914	0.7014
LSTM with GloVE (fine-tune)	0.001	1	0.9577	0.7153	0.7308

Best model: The higher learning rate model from HW3 that used the one-hot encoding instead of the word embedding layer outperformed all the other models experimented in Hw4. This model had the highest test accuracy of 0.7832 and highest train accuracy of 0.9596 which was greater than that of all the other models as can be seen by Table 3. Therefore, the higher learning rate model was also more generalizable on unseen data compared to the other models.

Discussion:

The LSTM with word embeddings performed better as it overfitted slowly. This indicates that learning embeddings along with classification was helpful. The GloVE embeddings (frozen) model did not generalize well to the dataset as can be seen by the lower test accuracy of 0.7014 in Table 3. It was less generalized to this dataset as its style differs from the dataset that the GloVE was trained on and the weights were frozen. The LSTM with GloVE (fine-tune) model performed better on unseen data than the model with frozen weights as can be seen by the higher training accuracy of 0.7308 in table 3.

Overfitting: All these models have validation and test accuracy less than train accuracy therefore overfitting is evident in all models.

Overall:

GloVE training data did not align well with the Tweets data therefore one-hot encoding model performed better than the models that used the GloVE word-embedding layers. Since one-hot encoding allows the models to learn directly from the dataset, it is good when the dataset consists of rare words. Moreover, GloVE is more fitting to use if the tasks involve sematic understanding. For a classification task, one-hot encoding is appropriate. The pretrained weights in GloVE may cause a bias which is why training a embedding layer from scratch is better.

Suggestions:

Dropouts could have been used in embedding and LSTM layers to prevent overfitting and moreover L2 regularization could have been used to prevent the model from memorizing the given data. And use of data augmentation would have allowed for more data variability and would have helped to prevent overfitting. Maybe a different sequence length other than 30 could have been experimented with to yield better results.