Graded Quiz: Test your Project Understanding

Latest Submission Grade 100%

1.

Question 1

Which of the following models generate context-based representations of text? (Select all that apply)

1 / 1 point

☑

Embeddings from Language Models (ELMo)

Correct

Correct! ELMo uses character-based word representations and bidirectional LSTMs. The pre-trained model computes a contextualised vector of 1024 dimensions.

□

GloVe

☑

Universal Sentence Encoder (USE)

Correct

Correct. USE uses a Transformer architecture that uses an attention mechanism to incorporate information about the order and the collection of words. The pre-trained model of USE that returns a vector of 512 dimensions is also available on Tensorflow Hub.

☑

Neural-Net Language Model (NNLM)

Correct

You got it! The model simultaneously learns representations of words and probability functions for word sequences, allowing it to capture semantics of a sentence.

In the project, we used pretrained models available on Tensorflow Hub, that are trained on the English Google News 200B corpus, and computed vectors of 128 dimensions for the larger model and 50 dimensions for the smaller model.

2.

Question 2

In the hands-on project, did we write code to preprocess the text? Please explain. Preprocessing steps can include removing stop words, stemming, lemmatization, tf-idf, tokenization, padding, etc.

1 / 1 point

○

Yes. The text corpus needed to be appropriately vectorized (converted into a numerical representation) for the classification models to be able to use them. Theorefore, we used text preprocessing modules from **tf.keras** to turn each text into a sequence of integers.

●

No. The text embedding modules we used from TF Hub preprocess the the input text. The modules do not require preprocessing the data before applying the modules, as preprocessing of input text is part of the TensorFlow graph.

Correct

Good job! For more, read the preprocessing section for the Universal Sentence Encoder and NNLM over on TF Hub.

3.

Question 3

Say you decide to build a binary text classification model to identify positive and negative sentiments associated with product reviews on Amazon. As you've taken this project, you have learned about using TensorFlow Hub and it's pre-traiined text embedding modules for NLP. How would you load TF Hub modules in your sequential module?

1 / 1 point

●

```
hub_layer = hub.KerasLayer("https://tfhub.dev/google/tf2-preview/nnlm-en-dim128/1", output_shape=[128],
                           input_shape=[], dtype=tf.string)

model = keras.Sequential()
model.add(hub_layer)
model.add(keras.layers.Dense(16, activation='relu'))
model.add(keras.layers.Dense(1, activation='sigmoid'))

model.summary()
```

```
hub_layer = hub.load("https://tfhub.dev/google/tf2-preview/nnlm-en-dim128/1", output_shape=[128],
                           input_shape=[], dtype=tf.string)

model = keras.Sequential()
model.add(hub_layer)
model.add(keras.layers.Dense(16, activation='relu'))
model.add(keras.layers.Dense(1, activation='sigmoid'))

model.summary()
```

```
model = keras.Sequential()

model.add("https://tfhub.dev/google/tf2-preview/nnlm-en-dim128/1", output_shape=[128], input_shape=[], dtype=tf.string)

model.add(keras.layers.Dense(16, activation='relu'))

model.add(keras.layers.Dense(1, activation='sigmoid'))


model.summary()
```

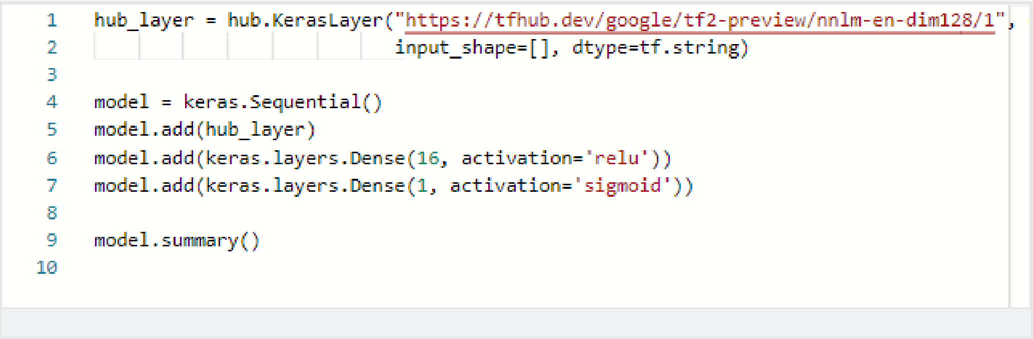
Correct

Correct! We did exactly this in Tasks 5 and 6.

3. Say you decide to build a binary text classification model to identify positive and negative sentiments associated with product reviews on Amazon. As you've taken this project, you have learned about using TensorFlow Hub and it's pre-traiined text embedding modules for NLP. How would you load TF Hub modules in your sequential module?

1 / 1 point

●



```
hub_layer = hub.KerasLayer("https://tfhub.dev/google/tf2-preview/nnlm-en-dim128/1",
                           input_shape=[], dtype=tf.string)

model = keras.Sequential()
model.add(hub_layer)
model.add(keras.layers.Dense(16, activation='relu'))
model.add(keras.layers.Dense(1, activation='sigmoid'))

model.summary()
```

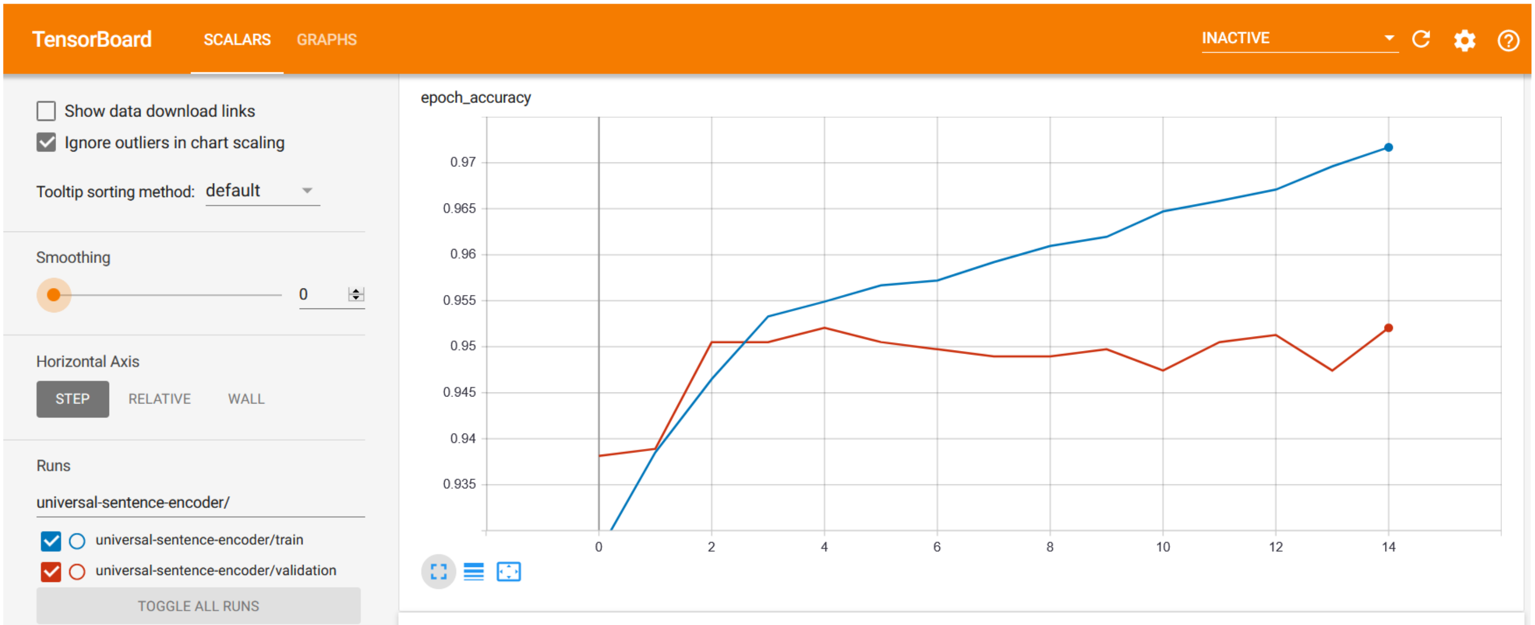
4.

Question 4

While visualizing your model's accuracy with Tensorboard, you obtain the following graph. The vertical axis represent model accuracy, while the horizontal axis represents the number of training epochs. The training accuracy and validation accuracy are given by the blue and red curves, respectively.

What do you think the model is doing?

1 / 1 point



☐ Show data download links

☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0

Horizontal Axis: STEP RELATIVE WALL

Runs

universal-sentence-encoder/

☑ ○ universal-sentence-encoder/train

☑ ○ universal-sentence-encoder/validation

○ Underfitting

● Overfitting

○ Coverged to a global optima

Correct

Correct! We see that the accuracy of our model on the validation data would peaks after training for a number of epochs, and then stagnates and starts decreasing. In other words, our model overfit to the training data. Learning how to deal with overfitting is important. Although it's often possible to achieve high accuracy on the training set, what we really want is to develop models that generalize well to a testing set (or data they haven't seen before).

5.

Question 5

Why should transfer learning work in NLP? Feel free to revisit the recordings and notebook. What are your intuitions, if any?

1 / 1 point

Transfer learning makes it possible to save training resources and to achieve good model generalization even when training on a small dataset.

Correct

Here are a few reasons I could think of:

- Many NLP tasks share common knowledge about language (linguistic representations, structural similarities, syntax, semantics...).
- Annotated data is rare, make use of as much supervision as possible. If you can combine data sets that you used for several tasks to get much bigger datasets. Bigger datasets are generally better for deep learning models.
- Unlabelled data is abundant (e.g. on the world wide web) and one should try to use as much of it as possible.
- Empirically, transfer learning has resulted in SOTA results for many supervised NLP tasks (e.g. classification, information extraction, Q&A, etc).