Week 1 Quiz

What is the name of the object used to tokenize sentences?

CharacterTokenizer

TextTokenizer

Tokenizer

WordTokenizer

What is the name of the method used to tokenize a list of sentences?
fit_on_texts(sentences)

tokenize(sentences)

fit_to_text(sentences)

tokenize_on_text(sentences)

Once you have the corpus tokenized, what's the method used to encode a list of sentences to use those tokens?

texts_to_tokens(sentences)

text_to_sequences(sentences)

texts_to_sequences(sentences)

text_to_tokens(sentences)

When initializing the tokenizer, how to you specify a token to use for unknown words?

unknown_word=<Token>

unknown_token=<Token>

out_of_vocab=<Token>

oov_token=<Token>

If you don't use a token for out of vocabulary words, what happens at encoding?

The word is replaced by the most common token

The word isn't encoded, and is skipped in the sequence

The word isn't encoded, and the sequencing ends

The word isn't encoded, and is replaced by a zero in the sequence

---

If you have a number of sequences of different lengths, how do you ensure that they are understood when fed into a neural network?

Process them on the input layer of the Neural Netword using the pad_sequences property

Specify the input layer of the Neural Network to expect different sizes with dynamic_length

Use the pad_sequences object from the tensorflow.keras.preprocessing.sequence namespace

Make sure that they are all the same length using the pad_sequences method of the tokenizer

If you have a number of sequences of different length, and call pad_sequences on them, what's the default result?

They'll get padded to the length of the longest sequence by adding zeros to the end of shorter ones

Nothing, they'll remain unchanged

They'll get padded to the length of the longest sequence by adding zeros to the beginning of shorter ones

They'll get cropped to the length of the shortest sequence

When padding sequences, if you want the padding to be at the end of the sequence, how do you do it?

Pass padding='post' to pad_sequences when initializing it

Pass padding='after' to pad_sequences when initializing it

Call the padding method of the pad_sequences object, passing it 'after'

Call the padding method of the pad_sequences object, passing it 'post'

Week 2 Quiz

What is the name of the TensorFlow library containing common data that you can use to train and test neural networks?

TensorFlow Data Libraries

There is no library of common data sets, you have to use your own

TensorFlow Data

TensorFlow Datasets

How many reviews are there in the IMDB dataset and how are they split?

50,000 records, 80/20 train/test split

50,000 records, 50/50 train/test split

60,000 records, 50/50 train/test split

60,000 records, 80/20 train/test split

How are the labels for the IMDB dataset encoded?

Reviews encoded as a number 1-5

Reviews encoded as a boolean true/false

Reviews encoded as a number 1-10

Reviews encoded as a number 0-1

What is the purpose of the embedding dimension?

It is the number of dimensions for the vector representing the word encoding

It is the number of words to encode in the embedding

It is the number of letters in the word, denoting the size of the encoding

It is the number of dimensions required to encode every word in the corpus

When tokenizing a corpus, what does the num_words=n parameter do?

It errors out if there are more than n distinct words in the corpus

It specifies the maximum number of words to be tokenized, and picks the first 'n' words that were tokenized

It specifies the maximum number of words to be tokenized, and picks the most common 'n' words

It specifies the maximum number of words to be tokenized, and stops tokenizing when it reaches n

To use word embeddings in TensorFlow, in a sequential layer, what is the name of the class?

tf.keras.layers.WordEmbedding

tf.keras.layers.Embed

tf.keras.layers.Word2Vector

tf.keras.layers.Embedding

IMDB Reviews are either positive or negative. What type of loss function should be used in this scenario?

Categorical crossentropy

Adam

Binary crossentropy

Binary Gradient descent

When using IMDB Sub Words dataset, our results in classification were poor. Why?

We didn't train long enough

Our neural network didn’t have enough layers

Sequence becomes much more important when dealing with subwords, but we’re ignoring word positions

The sub words make no sense, so can’t be classified

Week 3 Quiz

Why does sequence make a large difference when determining semantics of language?

It doesn’t

Because the order in which words appear dictate their impact on the meaning of the sentence

Because the order of words doesn’t matter

Because the order in which words appear dictate their meaning

How do Recurrent Neural Networks help you understand the impact of sequence on meaning?

They carry meaning from one cell to the next

They don’t

They look at the whole sentence at a time

They shuffle the words evenly

How does an LSTM help understand meaning when words that qualify each other aren’t necessarily beside each other in a sentence?

They shuffle the words randomly

They load all words into a cell state

Values from earlier words can be carried to later ones via a cell state

They don’t

What keras layer type allows LSTMs to look forward and backward in a sentence?

Bidirectional

Bothdirection

Bilateral

Unilateral

What’s the output shape of a bidirectional LSTM layer with 64 units?

(128,1)

(None, 128)

(128,None)

(None, 64)

When stacking LSTMs, how do you instruct an LSTM to feed the next one in the sequence?

Do nothing, TensorFlow handles this automatically

Ensure that return_sequences is set to True on all units

Ensure that they have the same number of units

Ensure that return_sequences is set to True only on units that feed to another LSTM

If a sentence has 120 tokens in it, and a Conv1D with 128 filters with a Kernal size of 5 is passed over it, what's the output shape?

(None, 116, 124)

(None, 120, 124)

(None, 120, 128)

(None, 116, 128)

What's the best way to avoid overfitting in NLP datasets?

Use LSTMs

Use GRUs

Use Conv1D

None of the above

Week 4 Quiz

What is the name of the method used to tokenize a list of sentences?

fit_on_texts(sentences)

fit_to_text(sentences)

tokenize_on_text(sentences)

tokenize(sentences)

If a sentence has 120 tokens in it, and a Conv1D with 128 filters with a Kernal size of 5 is passed over it, what's the output shape?

(None, 120, 124)

(None, 116, 124)

(None, 120, 128)

(None, 116, 128)

What is the purpose of the embedding dimension?

It is the number of dimensions for the vector representing the word encoding

It is the number of letters in the word, denoting the size of the encoding

It is the number of dimensions required to encode every word in the corpus

It is the number of words to encode in the embedding

IMDB Reviews are either positive or negative. What type of loss function should be used in this scenario?

Binary crossentropy

Adam

Categorical crossentropy

Binary Gradient descent

If you have a number of sequences of different lengths, how do you ensure that they are understood when fed into a neural network?

Process them on the input layer of the Neural Network using the pad_sequences property

Specify the input layer of the Neural Network to expect different sizes with dynamic_length

Use the pad_sequences object from the tensorflow.keras.preprocessing.sequence namespace

Make sure that they are all the same length using the pad_sequences method of the tokenizer

When predicting words to generate poetry, the more words predicted the more likely it will end up gibberish. Why?

Because the probability that each word matches an existing phrase goes down the more words you create

Because the probability of prediction compounds, and thus increases overall

It doesn't, the likelihood of gibberish doesn't change

Because you are more likely to hit words not in the training set

What is a major drawback of word-based training for text generation instead of character-based generation?

There is no major drawback, it's always better to do word-based training

Character based generation is more accurate because there are less characters to predict

Word based generation is more accurate because there is a larger body of words to draw from

Because there are far more words in a typical corpus than characters, it is much more memory intensive

How does an LSTM help understand meaning when words that qualify each other aren't necessarily beside each other in a sentence?

They shuffle the words randomly

Values from earlier words can be carried to later ones via a cell state

They load all words into a cell state

They don't