Kelime Gömme (Embeddings) ve Duygular

Bu colab dosyasında, sözcül yerleştirmeleriyle (gömmeleri) ile çalışacak ve metin duygularını tahmin etmek için temel bir sinir ağı eğiteceğiz.

TensorFlow'un ve Gereli İşlevlerin İçeri Aktarılması

In [1]:

```
import tensorflow as tf

from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
```

Veri Kümesinin Alınması

Amazon ve Yelp incelemelerini (yorumlar) ve ilgili duyarlılıklarıyla birlikte içeren bir veri seti kullanacağız (olumlu için 1, olumsuz için 0). Bu veri seti orijinal olarak buradan (https://www.kaggle.com/marklvl/sentiment-labelled-sentences-data-set) alınmıştır.

In [3]:

```
!wget --no-check-certificate \
    -O /tmp/sentiment.csv https://drive.google.com/uc?id=13ySLC_ue6Umt9RJYSeM2t-V0kCv-4C-P
```

```
--2021-07-28 09:00:23--  https://drive.google.com/uc?id=13ySLC_ue6Umt9RJYSeM
2t-V0kCv-4C-P (https://drive.google.com/uc?id=13ySLC_ue6Umt9RJYSeM2t-V0kCv-4
C-P)
Resolving drive.google.com (drive.google.com)... 74.125.142.101, 74.125.142.
102, 74.125.142.100, ...
Connecting to drive.google.com (drive.google.com)|74.125.142.101|:443... con
nected.
HTTP request sent, awaiting response... 302 Moved Temporarily
Location: https://doc-08-ak-docs.googleusercontent.com/docs/securesc/ha0ro93
7gcuc7l7deffksulhg5h7mbp1/qusduog65rn9doqokiguoabk09kvirn7/1627462800000/111
18900490791463723/*/13ySLC_ue6Umt9RJYSeM2t-V0kCv-4C-P (https://doc-08-ak-doc
s.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffksulhg5h7mbp1/qusdu
og65rn9doqokiguoabk09kvirn7/1627462800000/11118900490791463723/*/13ySLC_ue6U
mt9RJYSeM2t-V0kCv-4C-P) [following]
Warning: wildcards not supported in HTTP.
--2021-07-28 09:00:24--  https://doc-08-ak-docs.googleusercontent.com/docs/s
ecuresc/ha0ro937gcuc7l7deffksulhg5h7mbp1/qusduog65rn9doqokiguoabk09kvirn7/16
27462800000/11118900490791463723/*/13ySLC_ue6Umt9RJYSeM2t-V0kCv-4C-P (http
s://doc-08-ak-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffks
ulhg5h7mbp1/qusduog65rn9doqokiguoabk09kvirn7/1627462800000/11118900490791463
723/*/13ySLC_ue6Umt9RJYSeM2t-V0kCv-4C-P)
Resolving doc-08-ak-docs.googleusercontent.com (doc-08-ak-docs.googleusercon
tent.com)... 74.125.195.132, 2607:f8b0:400e:c09::84
Connecting to doc-08-ak-docs.googleusercontent.com (doc-08-ak-docs.googleuse
rcontent.com)|74.125.195.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 127831 (125K) [text/csv]
Saving to: ‘/tmp/sentiment.csv’

/tmp/sentiment.csv  100%[===================>] 124.83K  --.-KB/s    in 0.001
s

2021-07-28 09:00:24 (105 MB/s) - ‘/tmp/sentiment.csv’ saved [127831/127831]
```

In [4]:

```
import numpy as np
import pandas as pd

dataset = pd.read_csv('/tmp/sentiment.csv')

sentences = dataset['text'].tolist()
labels = dataset['sentiment'].tolist()

# Cümleleri ve etiketleri eğitim ve test setlerine ayırın
training_size = int(len(sentences) * 0.8)

training_sentences = sentences[0:training_size]
testing_sentences = sentences[training_size:]
training_labels = labels[0:training_size]
testing_labels = labels[training_size:]

# Etiketleri daha sonra ağda kullanmak üzere numpy dizileri haline getirin
training_labels_final = np.array(training_labels)
testing_labels_final = np.array(testing_labels)
```

Veri Kümesinin Tokenize Edilmesi

Doldurma ve `OOV` dahil olmak üzere veri kümesini tokenize edelim.

In [5]:

```
vocab_size = 1000
embedding_dim = 16
max_length = 100
trunc_type='post'
padding_type='post'
oov_tok = "<OOV>"


from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

tokenizer = Tokenizer(num_words = vocab_size, oov_token=oov_tok)
tokenizer.fit_on_texts(training_sentences)
word_index = tokenizer.word_index
sequences = tokenizer.texts_to_sequences(training_sentences)
padded = pad_sequences(sequences,maxlen=max_length, padding=padding_type,
                       truncating=trunc_type)

testing_sequences = tokenizer.texts_to_sequences(testing_sentences)
testing_padded = pad_sequences(testing_sequences,maxlen=max_length,
                               padding=padding_type, truncating=trunc_type)
```

Gözden Geçirilmesi

Yukarıdaki her şeyin uygun şekilde çalıştığından emin olmak için dolgulu dizilerden birine hızlıca göz atalım.

In [6]:

```
reverse_word_index = dict([(value, key) for (key, value) in word_index.items()])

def decode_review(text):
    return ' '.join([reverse_word_index.get(i, '?') for i in text])

print(decode_review(padded[1]))
print(training_sentences[1])
```

```
good case excellent value ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?
Good case Excellent value.
```

Gömmelerle Basit Bir Duygu Modeli Eğitelim

In [7]:

```
# Temel bir duygu ağı oluşturun
# Gömme katmanının ilk olduğuna ve çıktının 0 veya 1 (negatif veya pozitif) olduğundan yaln
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(vocab_size, embedding_dim, input_length=max_length),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(6, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 100, 16)           16000
_________________________________________________________________
flatten (Flatten)            (None, 1600)              0
_________________________________________________________________
dense (Dense)                (None, 6)                 9606
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 7
=================================================================
Total params: 25,613
Trainable params: 25,613
Non-trainable params: 0
_________________________________________________________________
```

In [8]:

```
num_epochs = 10
model.fit(padded, training_labels_final, epochs=num_epochs, validation_data=(testing_padded
```

```
Epoch 1/10
50/50 [==============================] - 4s 6ms/step - loss: 0.6926 - accura
cy: 0.5254 - val_loss: 0.6960 - val_accuracy: 0.4261
Epoch 2/10
50/50 [==============================] - 0s 4ms/step - loss: 0.6817 - accura
cy: 0.5298 - val_loss: 0.6906 - val_accuracy: 0.4737
Epoch 3/10
50/50 [==============================] - 0s 4ms/step - loss: 0.6513 - accura
cy: 0.5901 - val_loss: 0.6843 - val_accuracy: 0.4812
Epoch 4/10
50/50 [==============================] - 0s 4ms/step - loss: 0.5861 - accura
cy: 0.6874 - val_loss: 0.6558 - val_accuracy: 0.6040
Epoch 5/10
50/50 [==============================] - 0s 4ms/step - loss: 0.5098 - accura
cy: 0.8123 - val_loss: 0.6504 - val_accuracy: 0.5990
Epoch 6/10
50/50 [==============================] - 0s 4ms/step - loss: 0.4471 - accura
cy: 0.8751 - val_loss: 0.6354 - val_accuracy: 0.6617
Epoch 7/10
50/50 [==============================] - 0s 4ms/step - loss: 0.3959 - accura
cy: 0.9178 - val_loss: 0.6336 - val_accuracy: 0.6717
Epoch 8/10
50/50 [==============================] - 0s 4ms/step - loss: 0.3557 - accura
cy: 0.9485 - val_loss: 0.6217 - val_accuracy: 0.6817
Epoch 9/10
50/50 [==============================] - 0s 4ms/step - loss: 0.3247 - accura
cy: 0.9674 - val_loss: 0.6573 - val_accuracy: 0.6516
Epoch 10/10
50/50 [==============================] - 0s 4ms/step - loss: 0.2987 - accura
cy: 0.9768 - val_loss: 0.6589 - val_accuracy: 0.6692
```

Out[8]:

```
<tensorflow.python.keras.callbacks.History at 0x7fd8900dddd0>
```

Ağı Görselleştirmek İçin Dosyaları Alalım

Aşağıdaki kod, ağınızın her bir kelimeyle ilgili duyguyu nasıl gördüğünü görselleştirmek için iki dosya indirecektir. http://projector.tensorflow.org/ (http://projector.tensorflow.org/) adresine gidin ve bu dosyaları yükleyin, ardından "Sphereize" onay kutusunu tıklayın.

In [9]:

```
# İlk önce gömme katmanının ağırlıklarını alın
e = model.layers[0]
weights = e.get_weights()[0]
print(weights.shape) # şekil: (vocab_size, embedding_dim)
```

```
(1000, 16)
```

In [10]:

```
import io

# Gömme vektörlerini ve meta verileri yazın
out_v = io.open('vecs.tsv', 'w', encoding='utf-8')
out_m = io.open('meta.tsv', 'w', encoding='utf-8')
for word_num in range(1, vocab_size):
  word = reverse_word_index[word_num]
  embeddings = weights[word_num]
  out_m.write(word + "\n")
  out_v.write('\t'.join([str(x) for x in embeddings]) + "\n")
out_v.close()
out_m.close()
```

In [11]:

```
# Dosyaları İndirin
try:
  from google.colab import files
except ImportError:
  pass
else:
  files.download('vecs.tsv')
  files.download('meta.tsv')
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.Javascript object>
```

Yeni Yorumlar İle Duyguyu Tahmin Etme

Artık ağınızı eğittiğinize ve görselleştirdiğinize göre, ağın daha önce hiç görmediği yeni incelemelerde duyarlılığı nasıl tahmin edebileceğimize aşağıdaki kod bloğu ile bakabiliriz.

In [12]:

```
# Use the model to predict a review
fake_reviews = ['I love this phone', 'I hate spaghetti',
                'Everything was cold',
                'Everything was hot exactly as I wanted',
                'Everything was green',
                'the host seated us immediately',
                'they gave us free chocolate cake',
                'not sure about the wilted flowers on the table',
                'only works when I stand on tippy toes',
                'does not work when I stand on my head']

print(fake_reviews)

# Dizi oluşturun
padding_type='post'
sample_sequences = tokenizer.texts_to_sequences(fake_reviews)
fakes_padded = pad_sequences(sample_sequences, padding=padding_type, maxlen=max_length)

print('\nHOT OFF THE PRESS! HERE ARE SOME NEWLY MINTED, ABSOLUTELY GENUINE REVIEWS!\n')

classes = model.predict(fakes_padded)

# Sınıf 1'e ne kadar yakınsa, inceleme o kadar olumlu kabul edilir.
for x in range(len(fake_reviews)):
  print(fake_reviews[x])
  print(classes[x])
  print('\n')

# Kendi yorumlarınızı eklemeyi deneyin
# İyi incelemelere bazı olumsuz kelimeler ("değil" gibi) ekleyin ve ne olduğunu görün
# Örneğin:
# bize bedava çikolatalı kek verdiler ve bizden ücret almadılar
```

```
['I love this phone', 'I hate spaghetti', 'Everything was cold', 'Everything
was hot exactly as I wanted', 'Everything was green', 'the host seated us im
mediately', 'they gave us free chocolate cake', 'not sure about the wilted f
lowers on the table', 'only works when I stand on tippy toes', 'does not wor
k when I stand on my head']

HOT OFF THE PRESS! HERE ARE SOME NEWLY MINTED, ABSOLUTELY GENUINE REVIEWS!

I love this phone
[0.9859437]


I hate spaghetti
[0.39774713]


Everything was cold
[0.58342236]


Everything was hot exactly as I wanted
[0.77569896]


Everything was green
[0.54528093]
```

```
the host seated us immediately
[0.82662475]


they gave us free chocolate cake
[0.8717047]


not sure about the wilted flowers on the table
[0.39774713]


only works when I stand on tippy toes
[0.9464845]


does not work when I stand on my head
[0.39774713]
```