Cairo University

Faculty of Computers and Artificial Intelligence

Ass 4 NLP

Name	ID
1-Omer Salah	20200347
2-Mark Raouf	20190400
3-Mina Makram	20210603

Report of ass 4 CNN NLP :

About Dataset

-Context

This is the sentiment140 dataset. It contains 1,600,000 tweets extracted using the twitter api . The tweets have been annotated (0 = negative, 4 = positive) and they can be used to detect sentiment .

-Content

It contains the following 6 fields:

- target: the polarity of the tweet (0 = negative, 2 = neutral, 4 = positive)
- ids: The id of the tweet (2087)
- date: the date of the tweet (Sat May 16 23:58:44 UTC 2009)
- flag: The query (lyx). If there is no query, then this value is NO_QUERY.
- user: the user that tweeted (robotickilldozr)
- text: the text of the tweet (Lyx is cool)

We used only 2 fields of 6 fields [text - target]

Steps

1- load data

2- preprocessing
   - Remove tags
   - Smart lowercase
   - Remove numbers
   - Remove links
   - Remove Punctuation
   - Remove white spaces

3- Apply word2vec to get word embeddings.
   -Vocabulary size: 29578

4- Tokenizer
   -Total words 245208

5- Apply CNN model to classify tweets.

CNN Model 1 :

```
# Define CNN model
model = Sequential()

embedding_layer = Embedding(input_dim=vocab_size, output_dim=W2V_SIZE,
input_length=SEQUENCE_LENGTH, trainable=False)

embedding_layer.build((None,))
embedding_layer.set_weights([embedding_matrix])

model.add(embedding_layer)
model.add(Conv1D(filters=128, kernel_size=5, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(2, activation='softmax'))  # Adjust num_classes based on your classification
task

# Compile the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Print model summary
model.summary()
```

Test Accuracy for this model: 81.66093826293945

Test with 10 random samples from test set:

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
Sample 1:
  Actual Label: 1   Predicted Label: 1
Sample 2:
  Actual Label: 0   Predicted Label: 0
Sample 3:
  Actual Label: 0   Predicted Label: 0
Sample 4:
  Actual Label: 0   Predicted Label: 0
Sample 5:
  Actual Label: 1   Predicted Label: 1
Sample 6:
  Actual Label: 1   Predicted Label: 0
Sample 7:
  Actual Label: 1   Predicted Label: 1
Sample 8:
  Actual Label: 0   Predicted Label: 0
Sample 9:
  Actual Label: 0   Predicted Label: 1
Sample 10:
  Actual Label: 1   Predicted Label: 1
```

CNN Model 2 :

```
model = Sequential()

embedding_layer = Embedding(input_dim=vocab_size, output_dim=W2V_SIZE,
input_length=SEQUENCE_LENGTH, trainable=False)

embedding_layer.build((None,))
embedding_layer.set_weights([embedding_matrix])

model.add(embedding_layer)
model.add(Conv1D(filters=128, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))

# GlobalMaxPooling layer
model.add(GlobalMaxPooling1D())

# Dense layers
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))  # Adjust num_classes based on your classification
task

# Compile the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Print model summary
model.summary()
```

Test Accuracy for this model : 77.55687236785889

CNN Model 3 :

```
model = Sequential()

embedding_layer = Embedding(input_dim=vocab_size, output_dim=W2V_SIZE,
input_length=SEQUENCE_LENGTH, trainable=False)

embedding_layer.build((None,))
embedding_layer.set_weights([embedding_matrix])

model.add(embedding_layer)
model.add(Conv1D(filters=128, kernel_size=5, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=64, kernel_size=5, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=32, kernel_size=5, activation='relu'))
model.add(GlobalMaxPooling1D())

# Dense layers
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))
model.add(BatchNormalization())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))  # Adjust num_classes based on your classification
task

# Compile the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Print model summary
model.summary()
```

Test Accuracy for this model : 81.69281482696533