Preprocessign

Set up

In [ ]:
```
!pip install qalsadi
```

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
---------------------------------------------------------------------------
MessageError                              Traceback (most recent call last)
<ipython-input-3-d5df0069828e> in <module>
      1 from google.colab import drive
----> 2 drive.mount('/content/drive')

/usr/local/lib/python3.7/dist-packages/google/colab/drive.py in mount(mountpoint, force_remount, timeout_ms, re
adonly)
    104       timeout_ms=timeout_ms,
    105       ephemeral=True,
--> 106       readonly=readonly)
    107
    108

/usr/local/lib/python3.7/dist-packages/google/colab/drive.py in _mount(mountpoint, force_remount, timeout_ms, e
phemeral, readonly)
    123   if ephemeral:
    124     _message.blocking_request(
--> 125         'request_auth', request={'authType': 'dfs_ephemeral'}, timeout_sec=None)
    126
    127   mountpoint = _os.path.expanduser(mountpoint)

/usr/local/lib/python3.7/dist-packages/google/colab/_message.py in blocking_request(request_type, request, time
out_sec, parent)
    169   request_id = send_request(
    170       request_type, request, parent=parent, expect_reply=True)
--> 171   return read_reply_from_input(request_id, timeout_sec)

/usr/local/lib/python3.7/dist-packages/google/colab/_message.py in read_reply_from_input(message_id, timeout_se
c)
    100         reply.get('colab_msg_id') == message_id):
    101       if 'error' in reply:
--> 102         raise MessageError(reply['error'])
    103       return reply.get('data', None)
    104

MessageError: Error: credential propagation was unsuccessful
```

In [ ]:
```
import pandas as pd
import re
import qalsadi.lemmatizer
```

In [ ]:
```
project_dir = "/content/drive/MyDrive/afrisent-semeval-2023"
lang_code = "dz"
```

Creating the non-processed dataset

In [ ]:
```
## loading the data
import pandas as pd
df = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_train.tsv",sep="\t")
df = df.drop("ID",axis=1)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-dc2d64c932c0> in <module>
      1 ## loading the data
      2 import pandas as pd
----> 3 df = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_train.tsv",sep="\t")
      4 df = df.drop("ID",axis=1)

NameError: name 'project_dir' is not defined
```

In [ ]:
```
def f(x):
  if x == "positive":
    return 1
  elif x == "negative":
    return -1
  else:
    return 0
df.label = df.label.apply(lambda x:f(x))
```

In [ ]:
```
df.to_csv(f"{project_dir}/SubtaskA/train/{lang_code}_original.csv", index=False)
```

creating multiple varieties of preprocessed datasets

In [ ]:
```
# stopwords
! wget https://raw.githubusercontent.com/mohataher/arabic-stop-words/master/list.txt
```

In [ ]:
```
with open("list.txt", "r") as f:
  stopwords = [s.strip() for s in f.readlines()]
lemmer = qalsadi.lemmatizer.Lemmatizer()
punc = """،.:!؟!:.,''!"#$%&'()*+, -./:;<=>?@[\]^_`{|}~"""
```

In [ ]:
```
def preprocess(text):
    # removing @user amd RT
    text = text.replace("@user","").replace("RT","")
    # tokenization
    ara = re.findall(r'[\u0600-\u06FF]+', text)
    c = 0
    text = text.split()
    for i in range(len(text)):
      if len(re.findall(r'[\u0600-\u06FF]+', text[i])) > 0:
          text[i] = text[i].replace(ara[c], " " + ara[c] + " " )
          c  += 1
    text = " ".join(text).split()
    # # lemmatization
    # text = [lemmer.lemmatize(w) for w in text]
    # removing stopwords
    text = [w for w in text if not w in stopwords]
    # removing punctuation
    text = [w for w in text if not any(substring in w for substring in punc)]
    # removing numbers
    text = [w for w in text if not w.isdigit()]
    # normalizing emojis
    for i in range(len(text)):
        if len(re.findall(r'[\u0600-\u06FF]+', text[i])) == 0 and not text[i].isalnum():
            types = list(set(text[i]))
            del text[i]
            for j in range(len(types)):
              text.insert(i+j,types[j])
    # normalizing emojis second iteration
    for i in range(len(text)):
        if len(re.findall(r'[\u0600-\u06FF]+', text[i])) == 0 and not text[i].isalnum():
            types = list(set(text[i]))
            del text[i]
            for j in range(len(types)):
              text.insert(i+j,types[j])
    # removing empty strings
    text = [w for w in text if bool(w.strip())]
    return " ".join(text)
```

In [ ]:
```
pro_df = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_original.csv")
pro_df.tweet = pro_df.tweet.apply(lambda x:preprocess(x))
```

In [ ]:
```
pro_df.to_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro.csv", index=False)
```

In [ ]:
```
df = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro.csv")
df = df.dropna()
df = df.sample(frac=1).reset_index(drop=True)
df_test = pd.concat([df[df["label"] == -1][:60],df[df["label"] == 1][:60],df[df["label"] == 0][:60]],ignore_ind
df_train = df.drop(df_test.index)
df_train.to_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro_train.csv", index=False)
df_test.to_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro_test.csv", index=False)
```

LSTM

In [ ]:
```
!pip install tensorflow
!pip install keras
```

In [ ]:
```
from keras_preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense, Softmax, Dropout, Activation
from keras.layers import SimpleRNN, LSTM, Embedding, Bidirectional
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.preprocessing.text import Tokenizer
from keras.callbacks import ModelCheckpoint
import warnings
from keras.initializers import Constant
import tensorflow
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
```

```
project_dir = "/content/drive/MyDrive/afrisent-semeval-2023"
lang_code = "dz"
# aravec
# !wget https://bakrianoo.ewr1.vultrobjects.com/aravec/full_grams_sg_300_twitter.zip -P "/content/drive/MyDrive
# !unzip "/content/drive/MyDrive/afrisent-semeval-2023/full_grams_sg_300_twitter.zip" -d "/content/drive/MyDriv
import gensim
t_model = gensim.models.Word2Vec.load('/content/drive/MyDrive/afrisent-semeval-2023/full_grams_sg_300_twitter.m
```

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [ ]:
```
import pandas as pd
project_dir = "/content/drive/MyDrive/afrisent-semeval-2023"
lang_code = "dz"
df = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro.csv")
```

In [ ]:
```
df_train = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro_train.csv")
df_test = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro_test.csv")

x_train = df_train["tweet"]
y_train = df_train["label"]
x_test = df_test["tweet"]
y_test = df_test["label"]

encoder = Tokenizer(lower=False)
encoder.fit_on_texts(x_train)
x_train = encoder.texts_to_sequences(x_train)
x_test = encoder.texts_to_sequences(x_test)
total_words = len(encoder.word_index) + 1

def get_max_length():
    review_length = []
    for review in x_train:
        review_length.append(len(review))
    return int(np.ceil(np.mean(review_length)))
MAX_SEQUENCE_LENGTH=get_max_length()

from keras_preprocessing.sequence import pad_sequences
x_train = pad_sequences(x_train, maxlen=MAX_SEQUENCE_LENGTH, value=0, padding='post')
x_test = pad_sequences(x_test, maxlen=MAX_SEQUENCE_LENGTH, value=0, padding='post')

from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y_train=le.fit_transform(y_train)
y_test=le.transform(y_test)

# aravec
word_index = encoder.word_index
embedding_size=300
embedding_matrix = np.zeros((total_words, embedding_size))
for word, i in word_index.items():
    if word in t_model.wv:
        embedding_vector = t_model[word]
        embedding_matrix[i] = embedding_vector
```

In [ ]:
```
### Dziri bert embeddings
!pip install transformers
from transformers import pipeline
model = pipeline('feature-extraction', model= 'alger-ia/dziribert')
word_index = encoder.word_index
embedding_size=768
embedding_matrix = np.zeros((total_words, embedding_size))
for word, i in word_index.items():
        embedding_vector = model(word)[0][1]
        embedding_matrix[i] = embedding_vector
```

In [ ]:
```
model=Sequential()
model.add(Embedding(total_words,embedding_size,embeddings_initializer=Constant(embedding_matrix),input_length=M
model.add(LSTM(68, dropout = 0.5))
model.add(Dense(3,activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
checkpoint = ModelCheckpoint(f"{project_dir}/best_model.hdf5", monitor='val_accuracy', verbose=1,save_best_only
callback = tensorflow.keras.callbacks.EarlyStopping(monitor='loss', patience=4)
history= model.fit(x_train, to_categorical(y_train, num_classes=3), epochs=100,callbacks=[checkpoint,callback],
```

new Archtitecture

In [ ]:
```
df_train = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro_train.csv")
df_test = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro_test.csv")

# df_train["label"] = df_train["label"].apply(lambda x: 0 if(x==-1 or x==0) else 1)
# df_test["label"] = df_test["label"].apply(lambda x: 0 if(x==-1 or x==0) else 1)
```

```
x_train = df_train["tweet"]
y_train = df_train["label"]
x_test = df_test["tweet"]
y_test = df_test["label"]

encoder = Tokenizer(lower=False)
encoder.fit_on_texts(x_train)
x_train = encoder.texts_to_sequences(x_train)
x_test = encoder.texts_to_sequences(x_test)
total_words = len(encoder.word_index) + 1

def get_max_length():
    review_length = []
    for review in x_train:
        review_length.append(len(review))
    return int(np.ceil(np.mean(review_length)))
MAX_SEQUENCE_LENGTH=get_max_length()

from keras_preprocessing.sequence import pad_sequences
x_train = pad_sequences(x_train, maxlen=MAX_SEQUENCE_LENGTH, value=0, padding='post')
x_test = pad_sequences(x_test, maxlen=MAX_SEQUENCE_LENGTH, value=0, padding='post')

from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y_train=le.fit_transform(y_train)
y_test=le.transform(y_test)

# aravec
word_index = encoder.word_index
embedding_size=300
embedding_matrix = np.zeros((total_words, embedding_size))
for word, i in word_index.items():
    if word in t_model.wv:
        embedding_vector = t_model[word]
        embedding_matrix[i] = embedding_vector
```

In [ ]:
```
## Negative identifier
from keras.initializers import Constant
model=Sequential()
model.add(Embedding(total_words,300,embeddings_initializer=Constant(embedding_matrix),input_length=MAX_SEQUENCE
model.add(LSTM(8))
# model.add(Dense(256, activation = "sigmoid"))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
checkpoint = ModelCheckpoint(f"{project_dir}/best_pos_model.hdf5", monitor='val_accuracy', verbose=1,save_best_
callback = tensorflow.keras.callbacks.EarlyStopping(monitor='loss', patience=3)
history= model.fit(x_train, y_train, epochs=100,callbacks=[checkpoint,callback],validation_data=(x_test, y_test
```

In [ ]:
```
from keras.models import load_model
project_dir = "/content/drive/MyDrive/afrisent-semeval-2023"
neg_model = load_model(f"{project_dir}/best_neg_model.hdf5")
pos_model = load_model(f"{project_dir}/best_pos_model.hdf5")
```

In [ ]:
```
neg_train = neg_model.predict(x_train)
pos_train = pos_model.predict(x_train)
neg_test = neg_model.predict(x_test)
pos_test = pos_model.predict(x_test)
```

```
46/46 [==============================] - 0s 2ms/step
46/46 [==============================] - 0s 2ms/step
6/6 [==============================] - 0s 3ms/step
6/6 [==============================] - 0s 3ms/step
```

In [ ]:
```
import tensorflow
model=Sequential()
model.add(Dense(500, activation='relu'))
model.add(Dense(100, activation='relu'))
model.add(Dense(50, activation='relu'))
model.add(Dense(3,activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
checkpoint = ModelCheckpoint(f"{project_dir}/best_arch_model.hdf5", monitor='val_accuracy', verbose=1,save_best
callback = tensorflow.keras.callbacks.EarlyStopping(monitor='loss', patience=3)
history= model.fit(np.c_[neg_train,pos_train], to_categorical(y_train, num_classes=3), epochs=100,callbacks=[ch
```

In [ ]:
```
import pandas as pd
project_dir = "/content/drive/MyDrive/afrisent-semeval-2023"
lang_code = "dz"
df = pd.read_csv(f"{project_dir}/SubtaskA/train/{lang_code}_pro_train.csv")
df
```

Out[ ]:

	tweet	label
0	عنديش مزية كشعب نستحقوش النظافة النظام	-1
1	... زعما ننتا مول العقل أسي الغزواني هذاك راه حساب	-1
2	...يعمري غاضنتني بصح لبنات بلا استثناء واااو براف	-1
3	خدمات فاشلة تقول عاملين علينا مزية	-1
4	... اه علابالي الصحراء تقدر ترجعها جنة بصح المشكل	-1
...	...	...
1465	آش داني وعلاش مشيت هههههههه	0
1466	اخطونا بك بحكومتك بمسؤوليك	-1
1467	...العمرة ساعتين والحج تروح الصباح ترجع العشية ال	1
1468	يا اخي كنبغيك بزاف	1
1469	...التنظيم زعما غادي فمستوى عالي دورة وهران الالع	0

1470 rows × 2 columns