**1. الملخص التنفيذي**

هذا المشروع يهدف لبناء نظام بسيط لتحليل المشاعر باللغة العربية على مجموعة تغريدات. يجمع بين طريقتين:

1. تحليل معجمي (Dictionary-based) باستخدام قوائم كلمات موجبة وسالبة.
2. نموذج تعلم عميق (LSTM) يعتمد على تمثيل النصوص (Tokenization + Embedding).

الهدف: مقارنة الأداء، عرض النتائج بصريًا، وتقديم مقترحات لتحسين دقة النظام.

**2. أهداف المشروع**

* استخراج الاتجاه العام (سلب/إيجاب/محايد) من نصوص عربية.
* بناء قاعدة معالجة نصية قابلة لإعادة الاستخدام (تنظيف، تجزئة، تجذير).
* تدريب نموذج تصنيف وإظهار المقاييس الأساسية.
* توثيق خطوات التشغيل والتوصيات للخطوات القادمة.

**3. البيانات (Data)**

ملفات المشروع الأساسية:

* tweets.csv: يحتوي عمودًا باسم tweet يحوي النصوص.
* positive.csv، negative.csv: قاموس كلمات؛ يُفترض أن يحتوي كل ملف على عمود word وعمود polarity (القيمة العددية للدلالة على شدة القطبية).

**ملاحظات مهمة:**

* تأكد من ترميز الملفات (UTF-8) وخلوها من قيم مفقودة في الأعمدة الأساسية.
* إن كان عمود polarity في ملفات القواميس مُعطى كـ +1 أو -1 أو أعداد صحيحة، فتأكد من توحيد المعنى قبل الاستخدام.

**4. المعالجة المسبقة للنصوص (Preprocessing)**

الخطوات المطبقة في الكود مع توضيح سبب كل خطوة:

**4.1 تنظيف أساسي**

* إزالة الأرقام: re.sub(r'[0-9]+', '', text) — لأن الأرقام عادةً لا تضيف معنى شعوري.
* إزالة علامات الترقيم العربية والإنجليزية: قائمة arabic\_punctuations + string.punctuation.
* إزالة حروف التحكم (مثل LRI/RLI/FSI/PDI) لأنَّها تفسد عرض WordCloud وbidi.
* إزالة التكرارات المفرطة في الحرف: re.sub(r'(.)\1{2,}', r'\1', text) — لتقليل حالات مثل "هههههههه".

**4.2 التطبيع Normalization (مهم جدًّا للعربية)**

من الأفضل إضافة خطوات تطبيع مثل:

* تحويل جميع أشكال الألف (أ، إ، آ) إلى ا.
* تحويل ياء بأشكالها (ى → ي) وتهيئة التاء المربوطة (ة → ه أو ة حسب الاستخدام).
* إزالة الشدة والفتحة والكسرة (التشكيل).

مثال دالة تطبيع موجزة (يمكن إدراجها في الكود):

import re

def normalize\_arabic(text):

text = re.sub(r'[إأآا]', 'ا', text)

text = re.sub(r'ى', 'ي', text)

text = re.sub(r'ؤ', 'و', text)

text = re.sub(r'ئ', 'ي', text)

text = re.sub(r'ة', 'ه', text)

# إزالة التشكيل

text = re.sub(r'[\u0617-\u061A\u064B-\u0652]', '', text)

return text

**4.3 التجزئة، إزالة كلمات التوقف، والتجذير**

* word\_tokenize من NLTK: يعمل لكن قد لا يكون أدق للعربية؛ يمكن تجربة farasapy أو أدوات متخصصة.
* قائمة كلمات التوقف: مكتبة NLTK بها قائمة عربية، لكنها قد تكون ناقصة؛ أنصح بمراجعتها أو دمج قائمة مخصصة.
* التجذير: snowballstemmer موجود في المشروع كـ stemmer('arabic') — لكنه بسيط، واستخدام أدوات أقوى (مثل ISRI Stemmer أو lemmatizer أو Farasa) قد يحسن الأداء.

**5. بناء قاموس الكلمات (Dictionary Preprocessing)**

الملفات positive.csv وnegative.csv تُنظف بنفس أسلوب التغريدات:

* تنظيف الحقل word, حذف تكرارات، وعدم ترك قيم فارغة.
* إنتاج قاموسين dict\_positive وdict\_negative: مفتاح هو كلمة نظيفة، والقيمة هي polarity (عددية).

**ملاحظة:** تأكد من أن علامات polarity في السلبيات مُعطاة كقيمة سالبة (مثلاً -1) أو أن البرنامج يتعامل معها كقيمة سالبة؛ وإلا فستحتاج لتحويلها يدوياً.

**6. التحليل المعجمي (Dictionary-based Sentiment)**

الدالة sentiment\_analysis\_dict\_arabic(words\_list) تحسب مجموع القيم من القواميس:

score = sum(dict\_positive.get(word, 0) for word in words\_list) + sum(dict\_negative.get(word, 0) for word in words\_list)

ثم تُصنّف إلى positive/negative/neutral اعتمادًا على إشارة المجموع.

**ملاحظة فنية:** هذه الطريقة حساسة لوجود الكلمات في القواميس فقط، ولا تتعامل مع السياق (السلب المزدوج، النفي، السخرية، الخ).

**7. التمثيل البصري (Visualization)**

* **Pie chart**: عرض توزيع polarity الناتجة من التحليل.
* **WordCloud**: عند تكوين WordCloud للعربية يجب استخدام arabic\_reshaper.reshape ثم bidi.get\_display، مع خط يدعم الحروف العربية. كذلك إزالة أحرف التحكم قبل العرض.

**8. تجهيز البيانات للنموذج (Tokenization & Padding)**

* تحويل كل قائمة كلمات إلى جملة عن طريق toSentence.
* Tokenizer(num\_words=max\_words) ثم tokenizer.fit\_on\_texts(sentences).
* pad\_sequences(..., maxlen=max\_len) للحصول على مدخلات ثابتة الطول.
* الترميز: polarity\_encode = {'negative':0, 'neutral':1, 'positive':2}

**ملاحظة:** استخدم shuffle وstratify إذا كانت الفئات غير متوازنة.

**9. نموذج التعلم العميق (LSTM)**

البنية المستخدمة في المشروع:

* Embedding(input\_dim=max\_words, output\_dim=embed\_dim, input\_length=max\_len)
* LSTM(units=hidden\_unit, dropout=dropout\_rate)
* Dense(3, activation='softmax')

إعدادات التجميع: loss='sparse\_categorical\_crossentropy'، optimizer=Adam(...) أو RMSprop(...)، وmetrics=['accuracy'].

**تحسينات ممكنة:**

* استخدام Bidirectional(LSTM(...)) لتحسين التقاط السياق.
* إضافة recurrent\_dropout داخل LSTM.
* زيادة embed\_dim إلى 100–300 إذا استخدمت embeddings مُدربة.
* تجربة class\_weight أو oversampling/undersampling عند عدم توازن الفئات.

**10. التدريب والتقييم**

* تقسيم البيانات: train\_test\_split(X, y, test\_size=0.2, random\_state=0).
* التدريب: model.fit(..., validation\_split=0.1, epochs=10, batch\_size=256).
* المقاييس المحسوبة: Accuracy, Precision (weighted), Recall (weighted), F1 (weighted)، وConfusion Matrix.

**تحسينات عملية:**

* استخدام EarlyStopping(monitor='val\_loss', patience=3, restore\_best\_weights=True).
* استخدام ModelCheckpoint للحفظ التلقائي.
* تنفيذ stratify=y في train\_test\_split لحفظ توزيع الفئات.

**11. ملاحظات عملية على الكود الحالي (تحسينات وتصحيح أخطاء)**

1. **nltk.download('punkt\_tab')**: هذه قيمة غير صحيحة. المطلوب nltk.download('punkt') وnltk.download('stopwords') فقط.
2. تكرار import nltk مرتين في أعلى الكود — يمكن حذف التكرار.
3. المسارات المطلقة (D:\MyDesktop\...) تجعل الكود غير قابل للمشاركة بسهولة — استخدم مسارات نسبية أو اجعل المستخدم يمرر مسار البيانات كوسيط.
4. التحقق من أن positive.csv وnegative.csv تحتويان على عمود polarity بقيم مناسبة (سالب/موجب).
5. دالة remove\_repeating\_char قد تحوِّل كلمات مثل الله إلى اله. يجب تقييدها أو استثناء كلمات محددة أو تطبيقها فقط عندما يكون طول التكرار كبيرًا جدًا.
6. قبل إنشاء WordCloud تأكد من عدم وجود أحرف تحكم: تمت إضافة دالة remove\_control\_characters، وهي مفيدة.
7. إضافة طباعة عدد العينات بعد كل خطوة تنظيف ومخرجات لفحص التأثير (debugging).
8. التعامل مع القيم المفقودة (NaN) قبل التدريب وإزالة التغريدات الفارغة.
9. تثبيت Seed للتكرار (Reproducibility):

import numpy as np

import random

import tensorflow as tf

np.random.seed(0)

random.seed(0)

tf.random.set\_seed(0)

**12. مقترحات لتحسين الدقة والاعتماد**

* **استخدام Transformers**: نماذج مثل **AraBERT** أو **CAMeL-BERT** أو **MARBERT** تعطي نتائج أفضل بكثير من LSTM التقليدي على العربية.
* **استخدام تمثيلات مدربة مسبقًا** (FastText أو word2vec أو GloVe للعربية).
* **تحسين preprocessing**: خاصة تطبيع الحروف والتعامل مع النفي والتركيب الصرفي.
* **زيادة وتوسيع القواميس** (للمعجم) أو استخدام موارد خارجية مُعتمدة.
* **تقنيات Data Augmentation**: إضافة ترجمة عكسية (back-translation)، استبدال مرادفات.
* **تحليل أخطاء (Error Analysis)**: راجع أمثلة خطأ التصنيف لفهم الفشل.

**13. متطلبات النظام والمكتبات (requirements)**

قائمة مقترحة لـ requirements.txt:

python>=3.8

pandas

numpy

nltk

snowballstemmer

arabic\_reshaper

python-bidi

wordcloud

matplotlib

seaborn

scikit-learn

tensorflow

keras

أضف إصدارات محددة حسب بيئتك (مثلاً tensorflow==2.11.0) إذا رغبت في ثبات البيئة.

**14. دليل التشغيل خطوة بخطوة**

1. إنشاء بيئة افتراضية:

python -m venv venv

source venv/bin/activate # on mac/linux

venv\Scripts\activate # on Windows

1. تثبيت الحزم:

pip install -r requirements.txt

1. تنزيل موارد NLTK (مرّة واحدة):

import nltk

nltk.download('punkt')

nltk.download('stopwords')

1. تشغيل السكربت الرئيسي (بعد تعديل مسارات الملفات إن لزم):

python main.py

1. بعد التدريب سيُنتج ملف tweets\_clean\_polarity.csv ومخططات وتقييم في سطر الأوامر.

**15. كيفية وضع المحتوى في PowerPoint وWord**

* **لـ PowerPoint (3 شرائح):** انسخ كل شريحة من الجزء الأول (العنوان، النقاط، ملاحظات المتحدث) والصقها في شريحة منفصلة. ضع ملاحظات المتحدث في منطقة Notes داخل PowerPoint لتظهر أثناء العرض.
* **لـ Word:** انسخ الجزء الثاني (الوصف التفصيلي الكامل) والصقه في مستند Word. استخدم أنماط العناوين (Heading 1, Heading 2) للهيكلة واحتفظ بجداول أو أكواد كـ monospace.

**16. مخرجات متوقعة أمثلة**

* **مخطط دائري** يظهر نسب الإيجابي/السلبي/المحايد.
* **WordCloud** يبرز أكثر الكلمات تكرارًا بعد التجهيز.
* **مقاييس نموذج**: مثال افتراضي — Accuracy: 78%, Precision (weighted): 76%, Recall: 75%, F1: 75% (القيم تعتمد على البيانات الحقيقية).

**17. خاتمة**

المشروع هي قاعدة جيدة للبدء: يجمع بين تحليل بسيط بالمعجم ونموذج LSTM. للحصول على نتائج إنتاجية وموثوقة للعربية، أنصح بالتحول إلى نماذج مُسبقة التدريب (Transformers) وتحسين خطوات التطبيع والتجهيز.

بأشراف المهندس إبراهيم التعمري.