



Apis y Bot de Telegram

Youssef Ezzamarty



Índice

Manual de uso	3
Funcionamiento del código	3
Procesamiento del estado de ánimo	4
Procesamiento de lenguaje natural	6
Paso a python	14



Manual de uso

- **/start** para empezar.
- **/help** en caso de necesitar ayuda
- Escribe un mensaje describiendo tu estado de ánimo. El bot entiende el lenguaje natural en varios idiomas.

Funcionamiento de bot.php

1. Importa las dependencias necesarias usando `require 'vendor/autoload.php'` y use `GuzzleHttp\Client` para hacer solicitudes HTTP a la API de Telegram y la API de TMDb.
2. Define el token del bot de Telegram y la clave de API de TMDb.
3. Define una función llamada **enviarMensaje** que utiliza GuzzleHttp para enviar mensajes al chat de Telegram.
4. Define una función llamada **manejarMensajeNormal** que procesa el estado de ánimo del usuario, ejecuta un script de Python para analizar el estado de ánimo y luego utiliza la API de TMDb para obtener películas relacionadas.
5. Inicializa el cliente Guzzle HTTP.
6. Utiliza un bucle de polling para obtener actualizaciones de Telegram continuamente. Por cada actualización, verifica si es un mensaje nuevo y si es así, maneja el mensaje normalmente o envía un mensaje de bienvenida si es el comando **/start**.
7. Actualiza el ID del último mensaje procesado para evitar procesar mensajes antiguos repetidamente.
8. Espera un segundo antes de la próxima consulta para evitar un exceso de consumo de recursos.



Procesamiento del estado de ánimo

Uno de los atributos de la API de películas es ***genre_id***, algunos de los géneros más comunes y sus identificadores en TMDb son:

Acción: 28

Aventura: 12

Animación: 16

Comedia: 35

Crimen: 80

Documental: 99

Drama: 18

Familia: 10751

Fantasía: 14

Historia: 36

Terror: 27

Misterio: 9648

Romance: 10749

Ciencia ficción: 878

Televisión: 10770

Suspense: 53

Según el estado de ánimo seleccionamos algunos **genre_id** u otros.

```
// Obtener el estado de ánimo del mensaje
$estado_animo = $mensaje['text'];

// Ejecutar el script de Python con el estado de ánimo como argumento
$output = exec("python sentiments3.py \"$estado_animo\"");

// Definimos los géneros deseados
$genre_ids_array = [];

// Switch para seleccionar los géneros según el estado de ánimo
switch ($output) {
    case "feliz":
        $genre_ids_array = [28, 12, 16, 35];
        break;
    case "triste":
        $genre_ids_array = [10752, 18, 80, 10751];
        break;
    case "neutro":
        $genre_ids_array = [878, 12, 28, 27];
        break;
    case "contento":
        $genre_ids_array = [28, 12, 878];
        break;
    case "emocionado":
        $genre_ids_array = [18, 27, 53];
        break;
    case "entusiasmado":
        $genre_ids_array = [35, 10751, 10749];
        break;
    case "ansioso":
        $genre_ids_array = [53, 9648, 10749];
        break;
    default:
        enviarMensaje($mensaje['chat']['id'], "Estado de ánimo no reconocido");
        exit;
}
```

En el siguiente punto abordaremos cómo obtener el estado de ánimo.



Procesamiento de lenguaje natural

sentiments.py Usando `textblob`

```
1 import sys
2 from googletrans import Translator
3 from textblob import TextBlob
4
5 # Recibimos el texto desde PHP
6 texto_usuario = ' '.join(sys.argv[1:]) # Unimos todos los argumentos en una sola cadena
7
8 try:
9     # Creamos un objeto Translator de Google
10    translator = Translator()
11
12    # Traducimos el texto al inglés
13    translated_text = translator.translate(texto_usuario, dest='en').text
14
15    # Creamos un nuevo objeto TextBlob con el texto traducido
16    translated_blob = TextBlob(translated_text)
17
18    # Obtenemos el sentimiento del texto traducido
19    sentimiento = translated_blob.sentiment
20
21    # Convertimos el sentimiento detectado en un estado de ánimo
22    e_animo = ""
23    if sentimiento.polarity > 0.3:
24        e_animo = "feliz"
25    elif sentimiento.polarity < -0.3:
26        e_animo = "triste"
27    elif 0 <= sentimiento.polarity <= 0.3:
28        e_animo = "neutro"
29    elif 0.3 < sentimiento.polarity <= 0.5:
30        e_animo = "contento"
31    elif 0.5 < sentimiento.polarity <= 0.7:
32        e_animo = "emocionado"
33    elif 0.7 < sentimiento.polarity <= 0.9:
34        e_animo = "entusiasmado"
35    else:
36        e_animo = "ansioso"
37
38    # Devolvemos el estado de ánimo al script PHP
39    print(e_animo)
40
41 except Exception as e:
42     print("Error:", e)
43
```

Ventajas: Funciona correctamente.

Desventajas: Es poco preciso, menos preciso que los demás.



Usando un modelo previamente entrenado.

```
1  import sys
2  from googletrans import Translator
3  from transformers import pipeline
4
5  # Recibimos el texto desde PHP
6  texto_usuario = ' '.join(sys.argv[1:]) # Unimos todos los argumentos en una sola cadena
7
8  try:
9      # Creamos un objeto Translator de Google
10     translator = Translator()
11
12     # Traducimos el texto al inglés
13     translated_text = translator.translate(texto_usuario, dest='en').text
14
15     # Cargamos un modelo pre-entrenado para análisis de sentimientos
16     sentiment_analysis = pipeline("sentiment-analysis")
17
18     # Obtenemos el análisis de sentimientos del texto traducido
19     sentiment_result = sentiment_analysis(translated_text)[0]
20
21     # Extraemos la etiqueta de sentimiento y la probabilidad
22     label = sentiment_result['label']
23     score = sentiment_result['score']
24
25     # Convertimos el sentimiento detectado en un estado de ánimo
26     if label == 'POSITIVE':
27         if score > 0.7:
28             e_animo = "emocionado"
29         elif score > 0.5:
30             e_animo = "entusiasmado"
31         elif score > 0.3:
32             e_animo = "contento"
33         else:
34             e_animo = "feliz"
35     elif label == 'NEGATIVE':
36         e_animo = "triste"
37     else:
38         e_animo = "neutro"
39
40     # Devolvemos el estado de ánimo al script PHP
41     print(e_animo)
42
43 except Exception as e:
44     print("Error:", e)
45
```

Ventajas: Más preciso y personalizable.

Desventajas: Muy lento en comparación con los demás. Más difícil de gestionar, a veces devuelve lo que no debería.

Usando `vaderSentiment.vaderSentiment`

```
1 import sys
2 from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
3 from googletrans import Translator
4
5 # Recibimos el texto desde PHP
6 texto_usuario = ' '.join(sys.argv[1:]) # Unimos todos los argumentos en una sola cadena
7
8 try:
9     # Creamos un objeto Translator de Google
10    translator = Translator()
11
12    # Traducimos el texto al inglés
13    translated_text = translator.translate(texto_usuario, dest='en').text
14
15    # Creamos un analizador de sentimientos de VADER
16    analyzer = SentimentIntensityAnalyzer()
17
18    # Obtenemos el análisis de sentimientos del texto traducido
19    sentiment_scores = analyzer.polarity_scores(translated_text)
20
21    # Convertimos el sentimiento detectado en un estado de ánimo
22    compound_score = sentiment_scores['compound']
23
24    if compound_score > 0.7:
25        e_animo = "entusiasmado"
26    elif 0.5 <= compound_score <= 0.7:
27        e_animo = "emocionado"
28    elif 0.3 <= compound_score < 0.5:
29        e_animo = "contento"
30    elif -0.3 <= compound_score <= 0.3:
31        e_animo = "neutro"
32    elif -0.5 <= compound_score < -0.3:
33        e_animo = "triste"
34    elif compound_score < -0.7:
35        e_animo = "ansioso"
36    else:
37        e_animo = "feliz"
38
39    # Devolvemos el estado de ánimo al script PHP
40    print(e_animo)
41
42 except Exception as e:
43     print("Error:", e)
44
```

Ventajas: Funciona correctamente.

Desventajas: Al principio parecía que no funcionaba correctamente, sin embargo, ajustando el *switch case* da muy buenos resultados .



Comparaciones

Aquí se ejecutan los distintos scripts para ir comparando los tres programas de python y ajustando los algoritmos para obtener resultados adecuados. Estas bibliotecas de NLP (Natural Language Processing) de Python usan un *score* para determinar la polaridad de un mensaje. Según esta polaridad podemos saber si un texto es considerado que expresa un mensaje más positivo o negativo.

```
PS C:\xampp\htdocs\bottelegram> python .\sentiments.py Me siento un poco abrumado  
ansioso  
● PS C:\xampp\htdocs\bottelegram> python .\sentiments2.py Me siento un poco abrumado  
entusiasmado  
● PS C:\xampp\htdocs\bottelegram> python .\sentiments3.py Me siento un poco abrumado  
neutro  
○ PS C:\xampp\htdocs\bottelegram> █
```

```
neutro  
● PS C:\xampp\htdocs\bottelegram> python .\sentiments.py estoy un poco agotado  
ansioso  
● PS C:\xampp\htdocs\bottelegram> python .\sentiments2.py estoy un poco agotado  
emocionado  
● PS C:\xampp\htdocs\bottelegram> python .\sentiments3.py estoy un poco agotado  
neutro  
PS C:\xampp\htdocs\bottelegram> █
```

```
PS C:\xampp\htdocs\bottelegram> python .\sentiments3.py estoy muy mal  
feliz  
● PS C:\xampp\htdocs\bottelegram> python .\sentiments2.py estoy muy mal  
emocionado  
● PS C:\xampp\htdocs\bottelegram> python .\sentiments.py estoy muy mal  
triste  
○ PS C:\xampp\htdocs\bottelegram> █
```



Comparación final

Debido a la falta de precisión del primer script, optamos por comparar sentiments2.py y sentiments3.py después de haber realizado algunos ajustes previamente.

Con modelo entrenado previamente:

```
PS C:\xampp\htdocs\bottelegram> python .\sentiments2.py Me siento bien
entusiasmado I feel good 0.9998502731323242
PS C:\xampp\htdocs\bottelegram> python .\sentiments2.py Me siento un poco abrumado
entusiasmado I feel a little overwhelmed 0.9863094091415405
PS C:\xampp\htdocs\bottelegram> python .\sentiments2.py Me siento fatal
entusiasmado I feel terrible 0.9995606541633606
PS C:\xampp\htdocs\bottelegram> python .\sentiments2.py Tengo demasiadas ganas de ver una peli. Qué alegría
entusiasmado I am too wanting to see a movie.What a joy 0.9989701509475708
PS C:\xampp\htdocs\bottelegram> python .\sentiments2.py Hoy estoy triste
entusiasmado Today I am sad 0.9989007711410522
PS C:\xampp\htdocs\bottelegram>
```

Con la biblioteca python: `vaderSentiment.vaderSentiment`

```
PS C:\xampp\htdocs\bottelegram> python .\sentiments3.py Me siento bien
contento I feel good 0.4404
PS C:\xampp\htdocs\bottelegram> python .\sentiments3.py Me siento un poco abrumado
neutro I feel a little overwhelmed -0.024
PS C:\xampp\htdocs\bottelegram> python .\sentiments3.py Me siento fatal
mal I feel terrible -0.4767
PS C:\xampp\htdocs\bottelegram> python .\sentiments3.py Tengo demasiadas ganas de ver una peli. Qué alegría
emocionado I am too wanting to see a movie.What a joy 0.5859
PS C:\xampp\htdocs\bottelegram> python .\sentiments3.py Hoy estoy triste
mal Today I am sad -0.4767
PS C:\xampp\htdocs\bottelegram>
```

Finalmente ya podemos decidir qué script usaremos. En este caso el ganador parece el tercer código, que emplea `vaderSentiment.vaderSentiment`.



Algunos problemas que vamos a afrontar

- **Dependencias de Python:** Nos podemos encontrar con problemas a la hora de ejecutar los scripts de python, sobre todo relacionado a la biblioteca urllib3.
- **Caché:** Al ejecutar el bot este puede recibir una gran cantidad de mensajes y por lo tanto, devolver una gran cantidad de respuestas de forma descontrolada. Estos mensajes están guardados en la siguiente dirección. Estos se conocen como updates y sirven para ir actualizando el bot en tiempo real.

https://api.telegram.org/bot6554813207:AAGGK4LXVNO7CV6JM_EeOsdYbj_fuXsmzI/getUpdates

```
1 {
2   "ok": true,
3   "result": [
4     {
5       "update_id": 724570444,
6       "message": {
7         "message_id": 138,
8         "from": {
9           "id": 5283220867,
10          "is_bot": false,
11          "first_name": "Sousou",
12          "language_code": "en"
13        },
14        "chat": {
15          "id": 5283220867,
16          "first_name": "Sousou",
17          "type": "private"
18        },
19        "date": 1707491567,
20        "text": "/start",
21        "entities": [
22          {
23            "offset": 0,
24            "length": 6,
25            "type": "bot_command"
26          }
27        ]
28      }
29    ],
30    {
31      "update_id": 724570445,
32      "message": {
33        "message_id": 140,
34        "from": {
35          "id": 5283220867,
36          "is_bot": false,
37          "first_name": "Sousou",
38          "language_code": "en"
39        },
40        "chat": {
41          "id": 5283220867,
42          "first_name": "Sousou",
43          "type": "private"
44        },
45        "date": 1707491573,
46        "text": "me siento mal"
47      }
48    }
49  ]
50 }
```

Podemos hacer un script que reciba el JSON de `getUpdate` mediante el uso de la biblioteca `requests`, que hace peticiones GET de HTTP. A continuación extraemos el campo `'update_id'` y para finalizar usando la biblioteca mencionada anteriormente hacemos una petición a

[https://api.telegram.org/bot6554813207:AAGGK4LXVNO7CV6JM_EeOsdymbj_fuXsmzI/getupdates?offset=\[número_ID\]+1](https://api.telegram.org/bot6554813207:AAGGK4LXVNO7CV6JM_EeOsdymbj_fuXsmzI/getupdates?offset=[número_ID]+1)

Lo ejecutaremos antes de iniciar nuestro bot.

```
clearUpdates.py > ...
1  import requests
2
3  # URL base
4  base_url = "https://api.telegram.org/bot6554813207:AAGGK4LXVNO7CV6JM_EeOsdymbj_fuXsmzI/"
5  #GetUpdates
6  def get_updates():
7      url = base_url + "getUpdates"
8      response = requests.get(url)
9      if response.status_code == 200:
10         return response.json()
11     else:
12         print("Error:", response.status_code)
13         return None
14  updates = get_updates()
15  #Ver los update_id
16  def get_update_ids():
17      url = base_url + "getUpdates"
18      response = requests.get(url)
19      if response.status_code == 200:
20         updates = response.json()['result']
21         update_ids = [update['update_id'] for update in updates]
22         return update_ids
23     else:
24         print("Error:", response.status_code)
25         return None
26
27  def get_updates_with_offset(update_ids):
28      for update_id in update_ids:
29         url = base_url + f"getUpdates?offset={update_id + 1}"
30         # Send the GET request
31         response = requests.get(url)
32         if response.status_code == 200:
33             updates = response.json()['result']
34             print(updates)
35         else:
36             print("Error:", response.status_code)
37  # Call the function to get update IDs
38  update_ids = get_update_ids()
39  # Call the function to get updates with offset for each update ID
40  if update_ids is not None:
41      get_updates_with_offset(update_ids)
```



Paso del bot a Python

Debido a problemas de dependencias a la hora de usar PHP he preferido pasar a Python con un script que hace exactamente lo mismo. Además conseguimos un código más corto y legible, a su vez, seremos más rápidos a la hora de trabajar debido a que las bibliotecas son muy accesibles y fáciles de instalar.

```
bot.py > ...
1  import requests
2  import time
3  import subprocess
4  import os
5  from dotenv import load_dotenv
6
7  # Cargar variables de entorno desde el archivo .env
8  load_dotenv()
9
10 # Obtener el bot_token de las variables de entorno
11 bot_token = os.environ.get('bot_token')
12 api_key = os.environ.get('api_key')
13
14 last_processed_message_id = 0
15
16 def enviar_mensaje(chat_id, mensaje):
17     url = f"https://api.telegram.org/bot{bot_token}/sendMessage"
18     params = {'chat_id': chat_id, 'text': mensaje}
19     response = requests.get(url, params=params)
20     return response.json()
21
22 def manejar_mensaje_normal(mensaje):
23     global last_processed_message_id
24
25     estado_animo = mensaje['text']
26     output = subprocess.check_output(["python", "sentiments3.py", estado_animo]).decode().strip()
27
28     genre_ids_dict = {
29         "feliz": [28, 12, 16, 35],
30         "triste": [18, 80, 10749],
31         "neutro": [878, 12, 28, 27],
32         "contento": [28, 12, 878],
33         "emocionado": [18, 27, 53],
34         "mal": [99],
35         "entusiasmado": [35, 10751, 10749, 35],
36         "ansioso": [53, 9648, 10749, 27]
37     }
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

PS C:\xampp\htdocs\bottelegram> python bot.py