```
# Esta es la Parte 2 del código principal del TFG: N- Gramas.
# Esta sección complementa la Parte 1, donde se realiza el preprocesamiento de datos y en la Parte 3, el modelad
# Ambas partes han sido separadas por motivos de tamaño al subir el proyecto a GitHub, pero forman parte de un ú
# Es fundamental ejecutar previamente la Parte 1 en el mismo entorno para garantizar que todas las librerías est
# También existe una versión unificada del proyecto en formato .py, que incluye todo el proceso completo.
```

```
# 2. Análisis Descriptivo de N-Gramas

#Se lee el csv con los datos limpios obtenidos anteriormente y se filtra por la columna "lemmas" y se imprimen
df_analisis = pd.read_csv('Venezuela_limpia.csv')
dft= df_analisis['lemmas']
dft = [x for x in dft if str(x) != 'nan']
print(dft)
```

```
['work ago hospital aid refugee illegal migrant etc talk knowledge believe simply leave country apply asylu
```

```
#Se calcula el Valor de TF-IDF de los unigramas y se imprimen
tfIdfVectorizer=TfidfVectorizer(use_idf=True, ngram_range=(1,1))
tfIdf = tfIdfVectorizer.fit_transform(dft)
names=tfIdfVectorizer.get_feature_names_out()
freqs = tfIdf.sum(axis=0).A1
result= dict(zip(names, freqs))
print(result)
```

```
{'aaf': 2.086031264657588, 'aaron': 19.1996631762703, 'abandon': 11.55172128070215, 'abandoned': 9.03804691
```

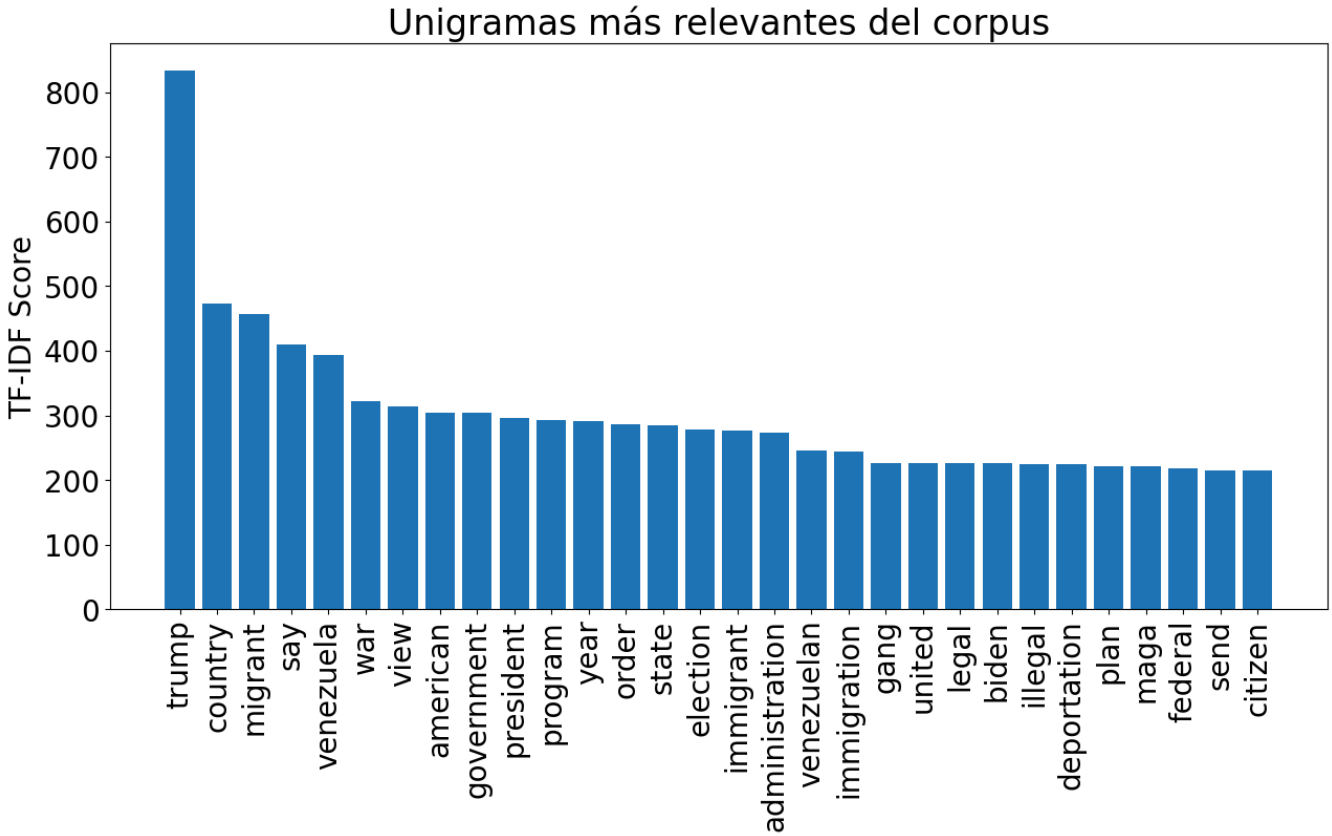
```
#Se muestran los 30 unigramas con mayor valor TF-IDF
from operator import itemgetter
i = 0
results_sorted=sorted(result.items(), key = itemgetter(1), reverse = True)
for key, value in results_sorted:
    i += 1
    if i == 31:
      break
    print(key, value)
```

```
trump 833.2672894012513
country 473.2925683694078
migrant 456.92181348648717
say 410.35222106152844
venezuela 392.9513829493716
war 322.081548453832
view 314.65202093736286
american 304.6842599540528
government 304.0827765914265
president 296.6339993892294
program 292.52470849999247
year 291.56553700069463
order 286.9823300344318
state 284.20108316112004
election 277.9183591286018
immigrant 275.89429487317886
administration 273.0571034761276
venezuelan 245.4059044995032
immigration 243.3658156107928
gang 227.04270963550232
united 226.86932795487775
legal 226.68776547678172
biden 226.02207221551703
illegal 225.30130049979275
deportation 224.6194836964085
plan 222.08087280383802
maga 220.74273281434742
federal 217.57070672953108
send 215.27206391958683
citizen 214.67380125166267
```

```
#Se pintan los 30 unigramas con más TF-IDF
df_results=pd.DataFrame.from_dict(results_sorted).head(30)
plt.rcParams.update({'font.size': 20})
plt.figure(figsize=(15,7))
```

```
plt.bar(df_results[0],df_results[1])
plt.xticks(rotation=90)
plt.ylabel('TF-IDF Score')
plt.title('Unigramas más relevantes del corpus')
```

```
Text(0.5, 1.0, 'Unigramas más relevantes del corpus')
```




```
#Se realiza la misma operación con los bigramas y trigramas

#Se calcula el Valor de TF-IDF de los bigramas
tfIdfVectorizer_bi=TfidfVectorizer(use_idf=True, ngram_range=(2,2))
tfIdf_bi = tfIdfVectorizer_bi.fit_transform(dft)
names_bi=tfIdfVectorizer_bi.get_feature_names_out()
freqs_bi = tfIdf_bi.sum(axis=0).A1
result_bi= dict(zip(names_bi, freqs_bi))
```

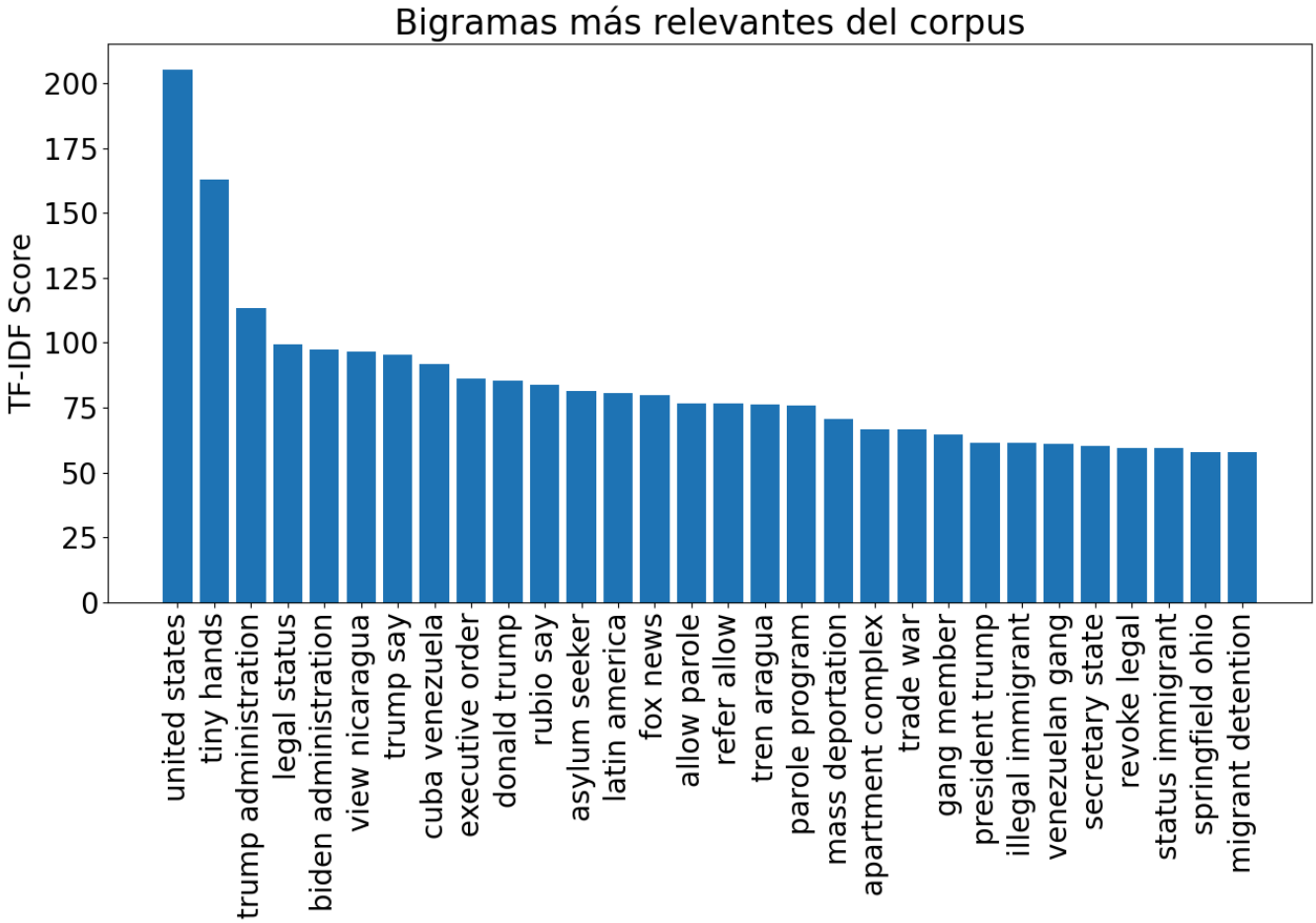
```
#Se muestran los 30 bigramas con mayor valor TF-IDF
from operator import itemgetter
i = 0
results_sorted=sorted(result_bi.items(), key = itemgetter(1), reverse = True)
for key, value in results_sorted:
    i += 1
    if i == 31:
      break
    print(key, value)
```

```
united states 205.01972240209133
tiny hands 162.97247694243288
trump administration 113.36255596699984
legal status 99.49776506477029
biden administration 97.17188057616056
view nicaragua 96.63395522502
trump say 95.53477068578523
cuba venezuela 91.7126530921621
executive order 85.99571213898318
donald trump 85.25428486537224
rubio say 83.81528940636791
asylum seeker 81.32344888698081
latin america 80.49475437698365
fox news 79.61969848522553
```

```
allow parole 76.76400617035398
refer allow 76.76400617035398
tren aragua 76.19946884087665
parole program 75.86339936989627
mass deportation 70.5325299536829
apartment complex 66.69657687061763
trade war 66.54403323639696
gang member 64.55597700589932
president trump 61.289606003191864
illegal immigrant 61.23377656650671
venezuelan gang 60.931930724282495
secretary state 60.35448695654123
revoke legal 59.2226084338374
status immigrant 59.2226084338374
springfield ohio 57.97586597434818
migrant detention 57.93857654523915
```

```
#Se pintan los 30 bigramas con más TF-IDF
results_sorted_bi=sorted(result_bi.items(), key = itemgetter(1), reverse = True)
df_results_bi=pd.DataFrame.from_dict(results_sorted_bi).head(30)
plt.rcParams.update({'font.size': 20})
plt.figure(figsize=(15,7))
plt.bar(df_results_bi[0],df_results_bi[1])
plt.xticks(rotation=90)
plt.ylabel('TF-IDF Score')
plt.title('Bigramas más relevantes del corpus')
```

```
Text(0.5, 1.0, 'Bigramas más relevantes del corpus')
```




```
#Trigramas
#Se calcula el Valor de TF-IDF de los trigramas
tfIdfVectorizer_tri=TfidfVectorizer(use_idf=True, ngram_range=(3,3))
tfIdf_tri = tfIdfVectorizer_tri.fit_transform(dft)
names_tri= tfIdfVectorizer_tri.get_feature_names_out()
freqs_tri = tfIdf_tri.sum(axis=0).A1
result_tri = dict(zip(names_tri, freqs_tri))
```

```
#Se muestran los 30 trigramas con mayor valor TF-IDF
from operator import itemgetter
i = 0
results_sorted=sorted(result_tri.items(), key = itemgetter(1), reverse = True)
for key, value in results_sorted:
    i += 1
    if i == 31:
      break
    print(key, value)
```

```
allow parole program 88.38834764831854
refer allow parole 88.38834764831854
legal status immigrant 57.861184090078154
revoke legal status 57.861184090078154
atlanta bureau chief 51.143831259375986
rick rojas atlanta 51.143831259375986
rojas atlanta bureau 51.143831259375986
sign executive order 47.481853992008986
member tren aragua 40.201291715270294
crazy cult leader 38.371058990949805
announcement reason end 37.68891807222048
condition dhs secretary 37.68891807222048
dhs secretary state 37.68891807222048
fleed well condition 37.68891807222048
migrant fleed well 37.68891807222048
reason end tps 37.68891807222048
secretary state announcement 37.68891807222048
state announcement reason 37.68891807222048
venezuela wave migrant 37.68891807222048
wave migrant fleed 37.68891807222048
well condition dhs 37.68891807222048
ally send special 36.084391824351506
close ally send 36.084391824351506
election talk invade 36.084391824351506
envoy nice maduro 36.084391824351506
maduro win election 36.084391824351506
nice maduro win 36.084391824351506
send special envoy 36.084391824351506
special envoy nice 36.084391824351506
start trade war 36.084391824351506
```

```
#Se pintan los 30 trigramas con más TF-IDF
results_sorted_tri=sorted(result_tri.items(), key = itemgetter(1), reverse = True)
df_results_tri=pd.DataFrame.from_dict(results_sorted_tri).head(30)
plt.rcParams.update({'font.size': 20})
plt.figure(figsize=(15,7))
plt.bar(df_results_tri[0],df_results_tri[1])
plt.xticks(rotation=90)
plt.ylabel('TF-IDF Score')
plt.title('Trigramas más relevantes del corpus')
```

```
Text(0.5, 1.0, 'Trigramas más relevantes del corpus')
```