Jacek Janur

Projekt na przedmiot PJN

Dane z [1]

Projekt polega na przypisaniu artykułowi z BBC na podstawie jego zawartości kategorii. W zbiorze danych mamy tylko 5 kategorii:

- sport
- business
- politics
- tech
- entertainment

W celach naukowych korzystamy z gotowego narzędzia TensorFlow Hub Universal Sentence Encoder. Jego krótki opis znajduje się w dalszej części pracy.

In [1]:
```
import numpy as np
import pandas as pd
from tensorflow import keras
import tensorflow as tf
import os
import tensorflow_hub as hub
import string
from sklearn.manifold import TSNE
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
from sklearn.preprocessing import LabelEncoder, LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import keras
from keras.models import Sequential
from keras.layers import Dense
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
```

Spróbujemy uruchomić notatnik na karcie graficznej aby przyśpieszyć obliczenia. Kod wzięty z [2]

In [2]:
```
try:
    tpu = tf.distribute.cluster_resolver.TPUClusterResolver()
    print('Running on GPU ', tpu.master())
except ValueError:
    tpu = None

if tpu:
    tf.config.experimental_connect_to_cluster(tpu)
    tf.tpu.experimental.initialize_tpu_system(tpu)
    strategy = tf.distribute.experimental.TPUStrategy(tpu)
else:
    strategy = tf.distribute.get_strategy()

print("REPLICAS: ", strategy.num_replicas_in_sync)
```

```
REPLICAS:  1
```

Wczytywanie i obsługa danych

In [3]:
```
df_train = pd.read_csv("bbc-text.csv")
df_train.describe()
```

Out[3]:

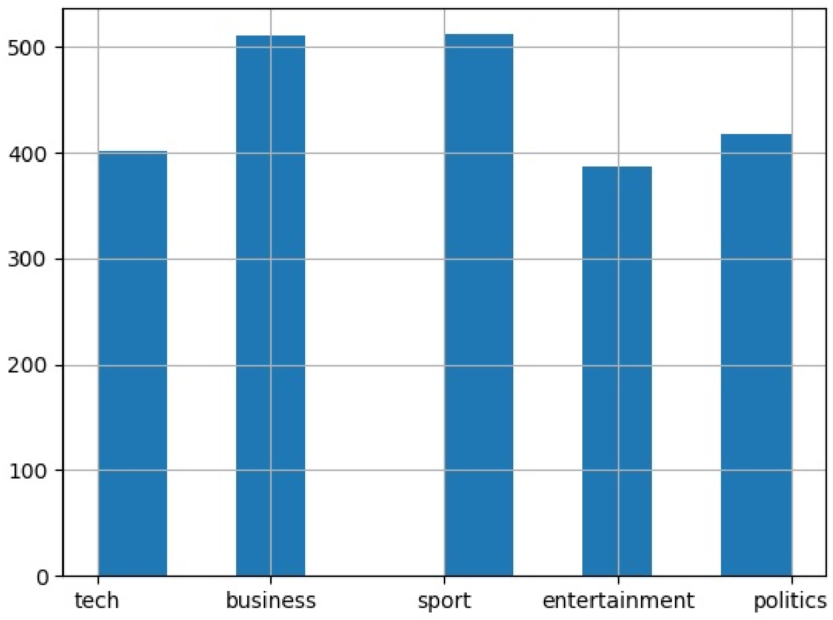
	category	text
count	2225	2225
unique	5	2126
top	sport	kennedy questions trust of blair lib dem leade...
freq	511	2

Utwórzmy histogram aby zobaczyć ile jest poszczególnych kategorii.

In [5]:
```
df_train["category"].hist()
```

Out[5]:

```
<Axes: >
```

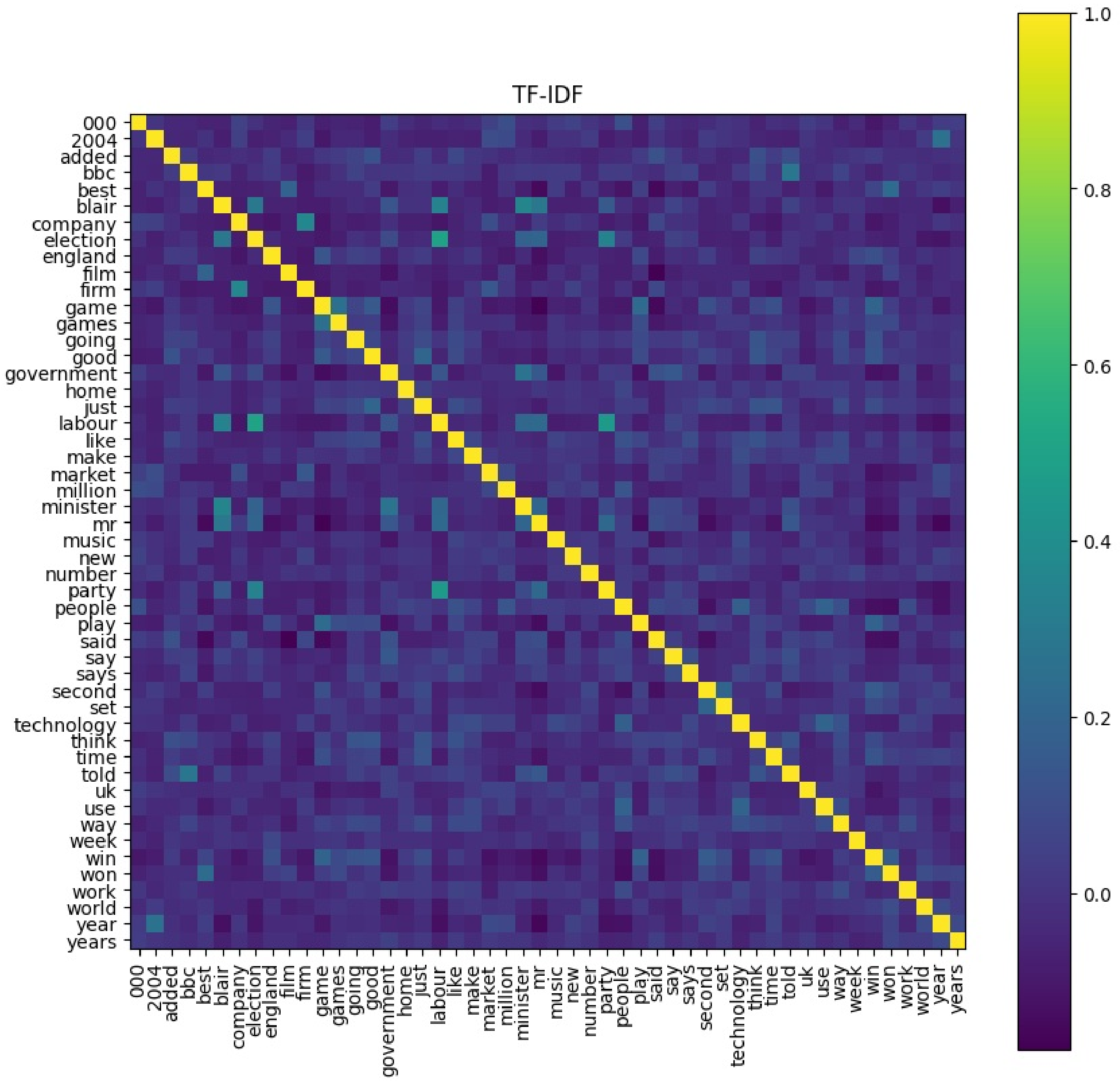


Term Frequency-Inverse Document Frequency (TF-IDF) Heatmaps

Jest to technika używana do oceny jak ważne jest słowo w danym dokumencie. Jeżeli dwa słowa mają jasny kolor, oznacza to, że często pojawiają się razem w dokumentach. Może to sugerować, że te słowa są powiązane tematycznie lub kontekstualnie.

In [18]:

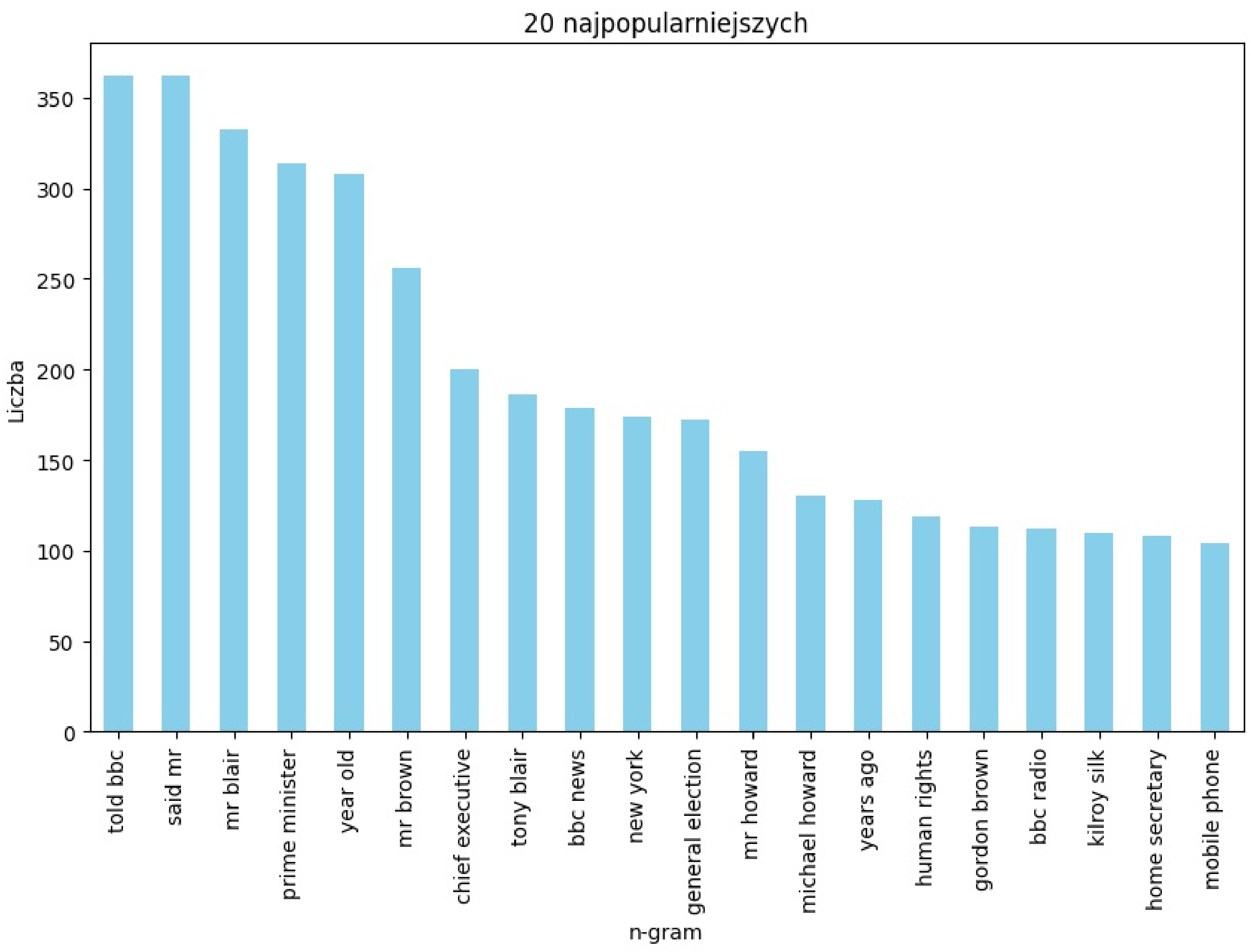
```
n = 50

# Inicjalizujemy TfidfVectorizer
vectorizer = TfidfVectorizer(max_features=n, stop_words='english')

# Przekształcamy nasz tekst
tfidf_matrix = vectorizer.fit_transform(df_train['text'])

# Tworzymy dataframe z wyników
df_tfidf = pd.DataFrame(tfidf_matrix.toarray(), columns=vectorizer.get_feature_names_out())

# Tworzymy korelację między cechami
corr = np.corrcoef(df_tfidf, rowvar=False)

# Tworzymy heatmapę
plt.figure(figsize=(10,10))
plt.imshow(corr, interpolation='none')
plt.colorbar()
plt.xticks(range(len(df_tfidf.columns)), df_tfidf.columns, rotation=90)
plt.yticks(range(len(df_tfidf.columns)), df_tfidf.columns)
plt.title('TF-IDF')
plt.show()
```



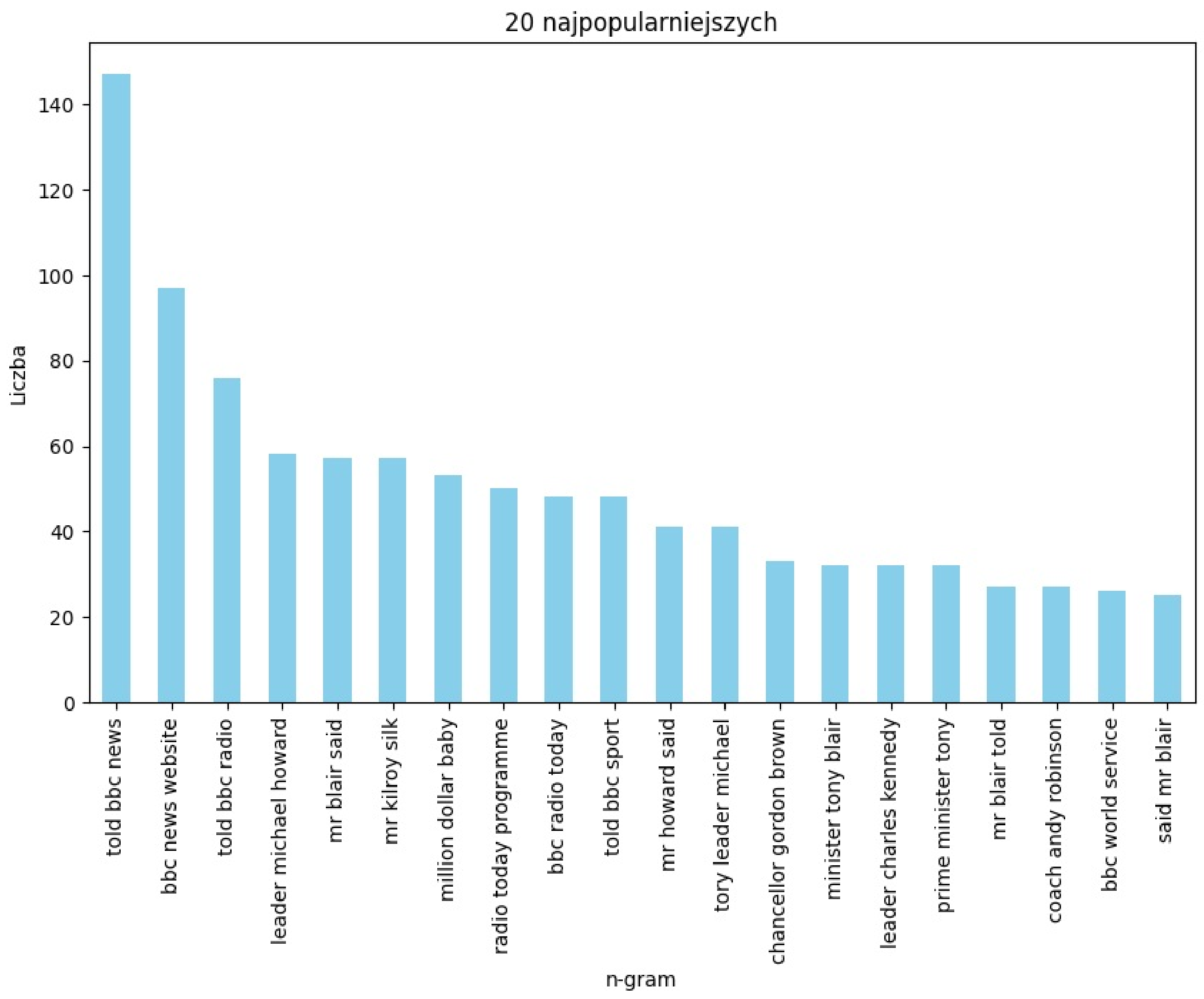
Wykresy n-gramów

Pokazują one, które grupy słów pojawiają się najczęściej razem. Pomaga to w wskazaniu tendencji w danym tekście.

In [19]:
```python
# Ustalamy liczbę n-gramów do analizy
n = 2

# Inicjalizujemy CountVectorizer
vectorizer = CountVectorizer(ngram_range=(n, n), stop_words='english')

# Przekształcamy nasz tekst
count_matrix = vectorizer.fit_transform(df_train['text'])

# Tworzymy dataframe z wyników
df_count = pd.DataFrame(count_matrix.toarray(), columns=vectorizer.get_feature_names_out())

# Sumujemy wystąpienia każdego n-gramu i sortujemy
df_count_totals = df_count.sum(axis=0).sort_values(ascending=False)

# Tworzymy wykres słupkowy
df_count_totals[:20].plot(kind='bar', figsize=(10,6), color='skyblue')
plt.title('20 najpopularniejszych')
plt.xlabel('n-gram')
plt.ylabel('Liczba')
plt.show()
```



In [16]:
```
# Ustalamy liczbę n-gramów do analizy
n = 3

# Inicjalizujemy CountVectorizer
vectorizer = CountVectorizer(ngram_range=(n, n), stop_words='english')

# Przekształcamy nasz tekst
count_matrix = vectorizer.fit_transform(df_train['text'])

# Tworzymy dataframe z wyników
df_count = pd.DataFrame(count_matrix.toarray(), columns=vectorizer.get_feature_names_out())

# Sumujemy wystąpienia każdego n-gramu i sortujemy
df_count_totals = df_count.sum(axis=0).sort_values(ascending=False)

# Tworzymy wykres słupkowy
df_count_totals[:20].plot(kind='bar', figsize=(10,6), color='skyblue')
plt.title('20 najpopularniejszych')
plt.xlabel('n-gram')
plt.ylabel('Liczba')
plt.show()
```

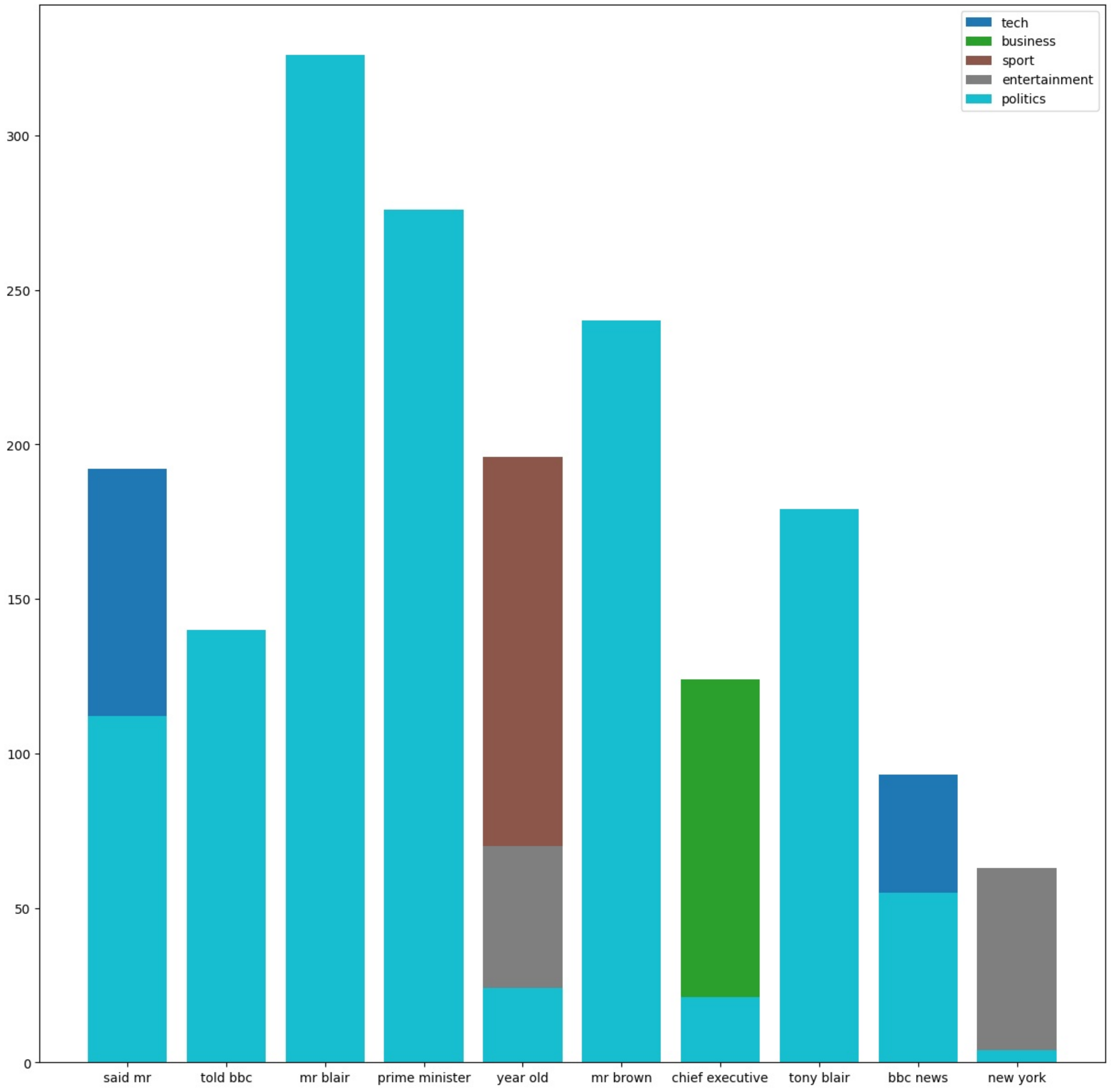


Co związku z tym?

Otrzymaliśmy najpopularniejsze 2, 3 - gramy dla posiadanego przez nas zbioru artykułów. Spróbujmy pokolorować odpowiednio n-gramy na podstawie kategorii gdzie występują najczęściej aby dowiedzieć się o nich więcej

In [14]:
```python
# Utworzenie n-gramów
def create_ngrams(df, n=2):
    count_vectorizer = CountVectorizer(ngram_range=(n, n), stop_words='english')
    count_vectorizer.fit(df['text'])
    return count_vectorizer

# Ustalamy liczbę n-gramów do analizy
n = 2
# Utworzenie obiektu CountVectorizer do tworzenia n-gramów
count_vectorizer = create_ngrams(df_train, n)

# Obliczenie częstości występowania n-gramów dla każdej kategorii
df_train['ngrams'] = df_train['text'].apply(lambda x: count_vectorizer.transform([x]).toarray()[0])
grouped = df_train.groupby('category')['ngrams'].apply(sum)

# Utworzenie DataFrame z n-gramami i ich częstościami
df_ngrams = pd.DataFrame(grouped.tolist(), index=grouped.index, columns=count_vectorizer.get_feature_names_out(

# Obliczenie najpopularniejszej kategorii dla każdego n-gramu
df_ngrams['most_common_category'] = df_ngrams.idxmax(axis=0)

# Obliczenie 20 najczęściej występujących n-gramów
most_common_ngrams = df_ngrams.sum(axis=0).astype(float).nlargest(10).index

# Utworzenie DataFrame z 20 najczęściej występującymi n-gramami i ich najpopularniejszymi kategoriami
df_most_common_ngrams = df_ngrams.loc[:, most_common_ngrams]

# Utworzenie mapy kolorów dla kategorii
categories = df_train['category'].unique()
colors = plt.get_cmap('tab10', len(categories))

# Utworzenie wykresu
fig, ax = plt.subplots()
```

```
for i, category in enumerate(categories):
    ax.bar(df_most_common_ngrams.columns, df_most_common_ngrams.loc[category, :], color=colors(i), label=catego

ax.legend()
fig.set_figheight(15)
fig.set_figwidth(15)
plt.show()
```



Jak możemy zauważyć najwięcej bigramów występuje w polityce. Od razu widać, że polityk Tony Blair był bardzo popularny w latach 2004-2005.

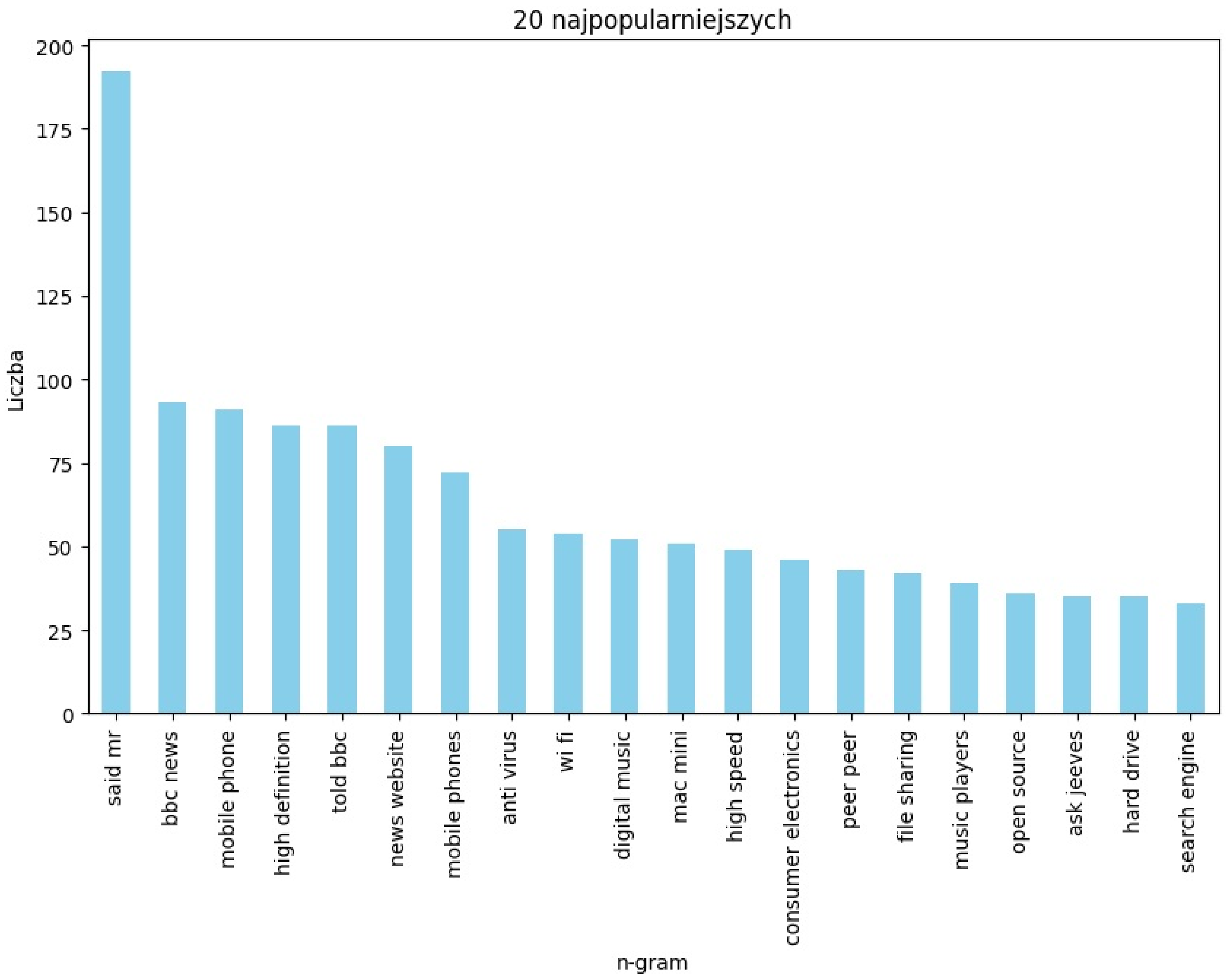
Stwórzmy jeszcze 2,3 - gramy dla każdej kategorii

Powtórzę powyższe kroki (pomijając ostatni z kolorowaniem) dla każdej kategorii aby lepiej zwizualizować dane.

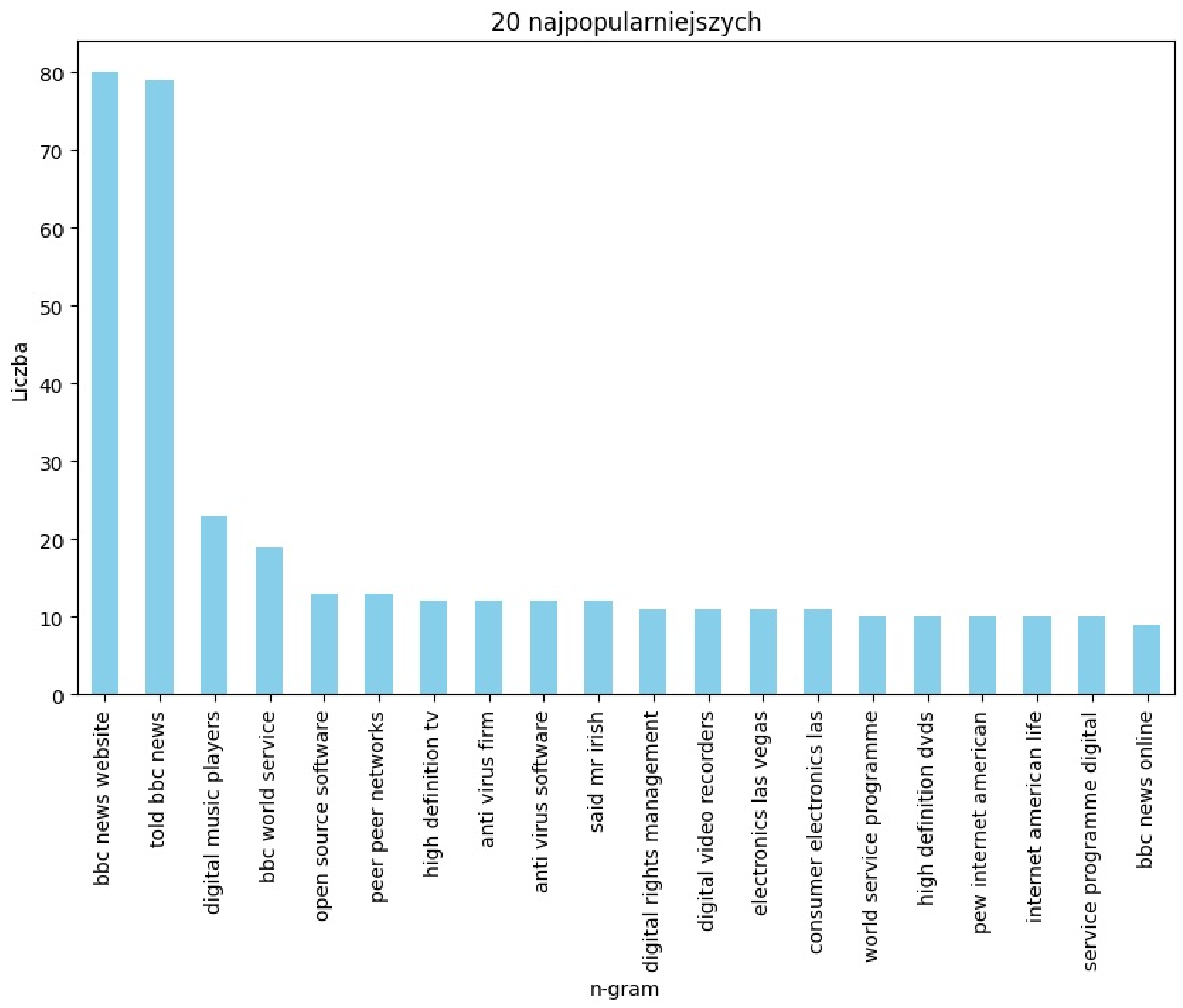
In [18]:
```
def show_ngram(df, n=2):
    vectorizer = CountVectorizer(ngram_range=(n, n), stop_words='english')

    # Przekształcamy nasz tekst
    count_matrix = vectorizer.fit_transform(df['text'])

    # Tworzymy dataframe z wyników
    df_count = pd.DataFrame(count_matrix.toarray(), columns=vectorizer.get_feature_names_out())

    # Sumujemy wystąpienia każdego n-gramu i sortujemy
    df_count_totals = df_count.sum(axis=0).sort_values(ascending=False)

    # Tworzymy wykres słupkowy
    df_count_totals[:20].plot(kind='bar', figsize=(10,6), color='skyblue')
    plt.title('20 najpopularniejszych')
    plt.xlabel('n-gram')
    plt.ylabel('Liczba')
    plt.show()
```

Tech

In [19]:
```
show_ngram(df_train[df_train['category'] == 'tech'], 2)
```

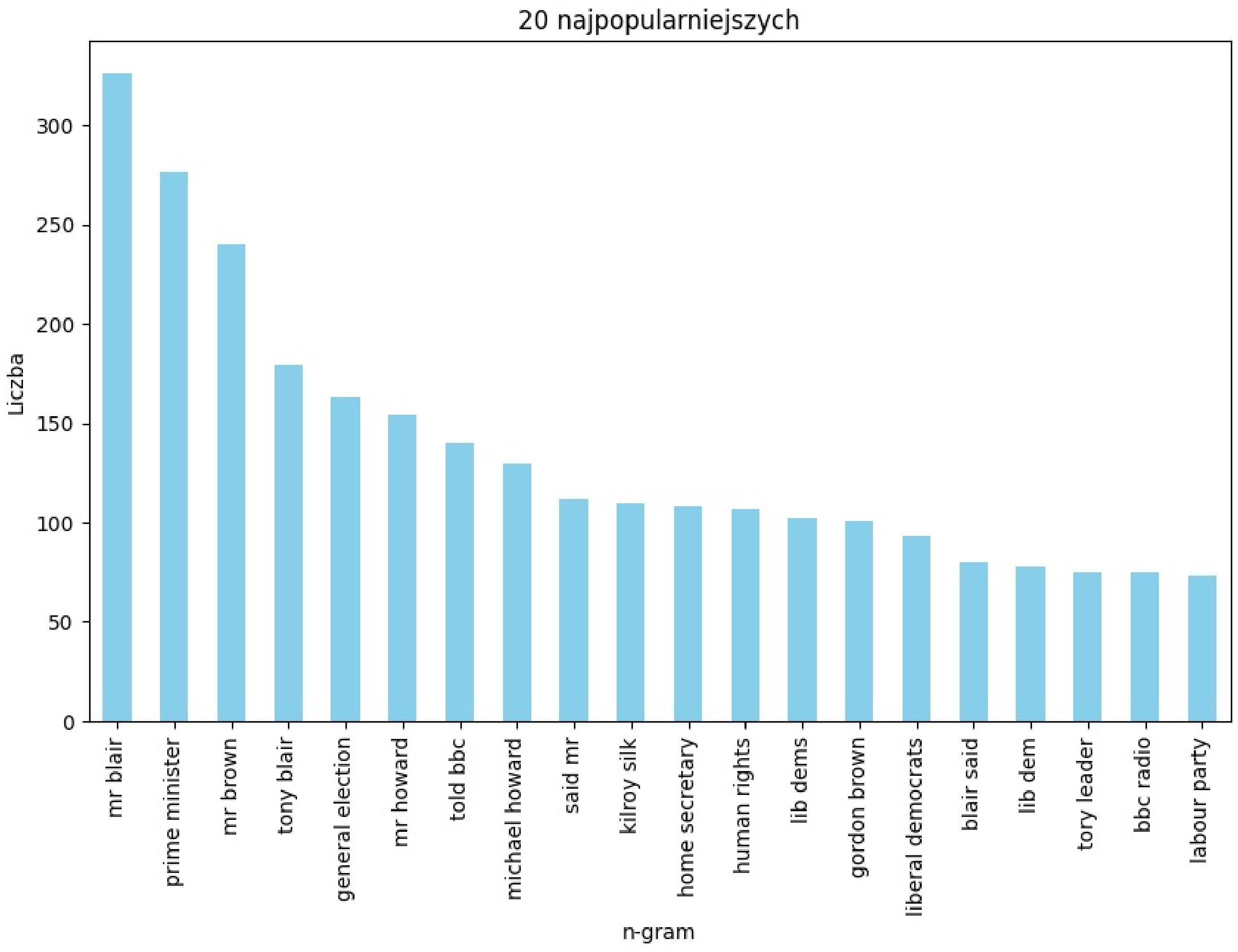


In [20]:
```
show_ngram(df_train[df_train['category'] == 'tech'], 3)
```

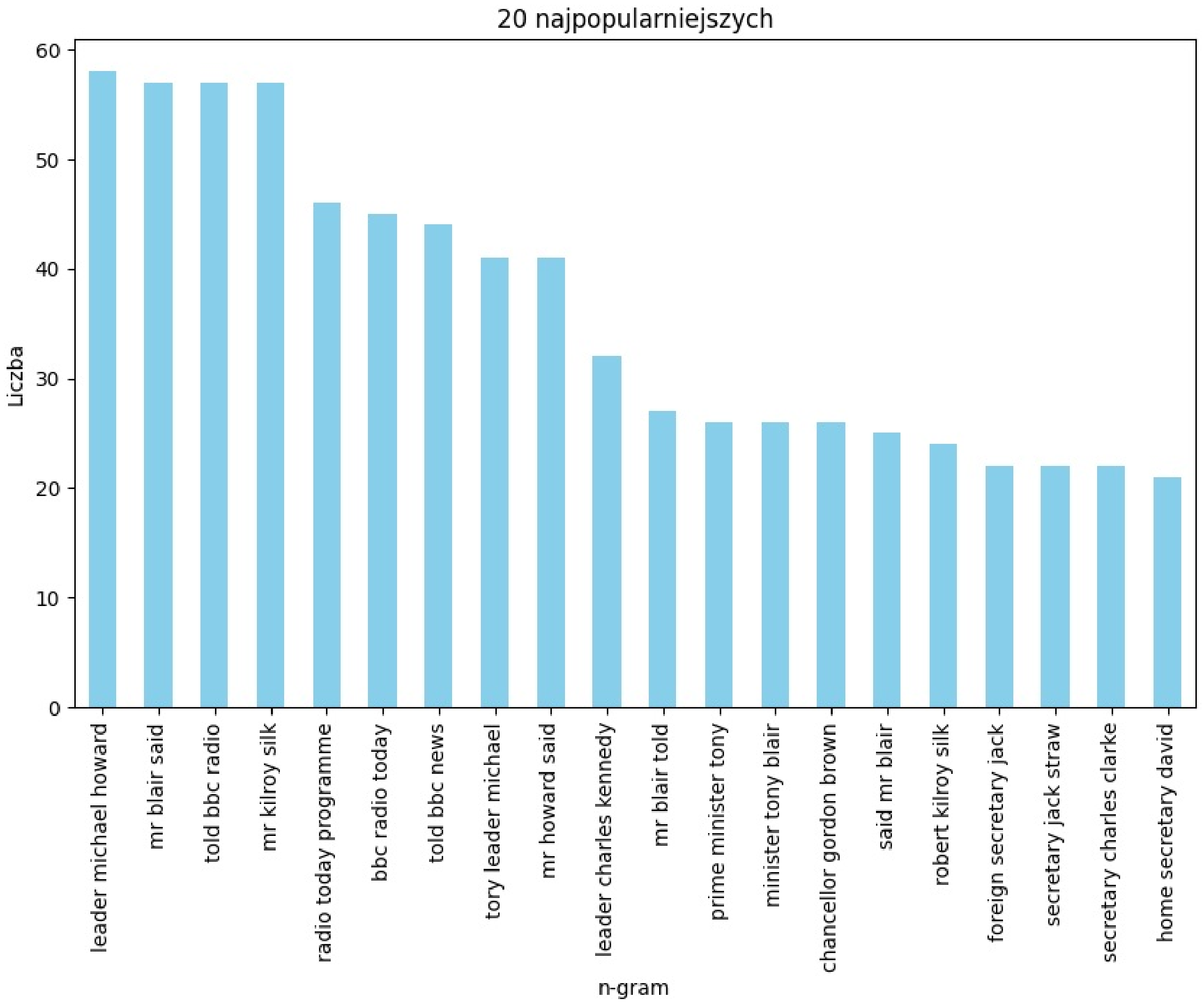


Politics

In [21]:
```
show_ngram(df_train[df_train['category'] == 'politics'], 2)
```

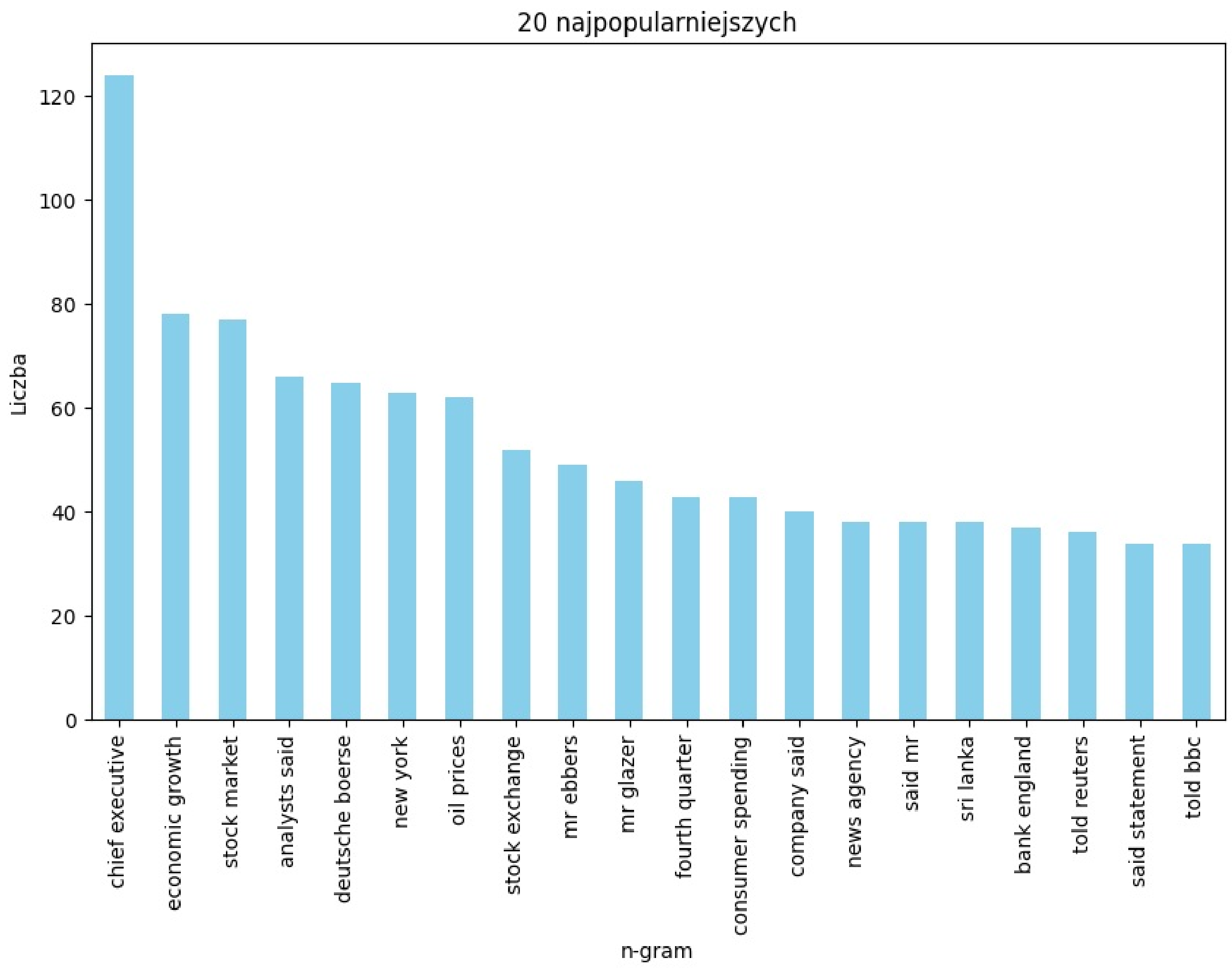


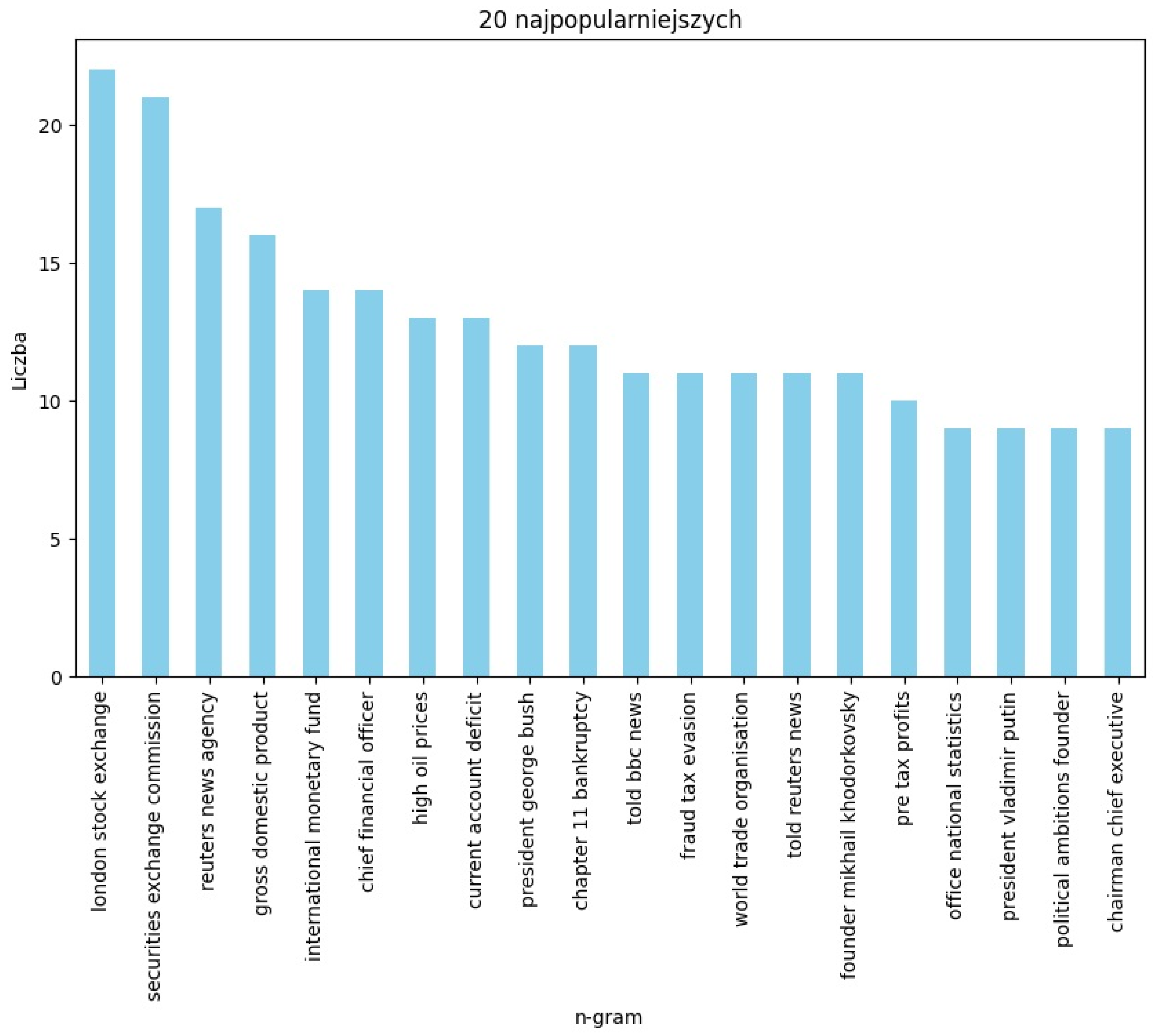
```
In [22]: show_ngram(df_train[df_train['category'] == 'politics'], 3)
```



Business

```
In [24]: show_ngram(df_train[df_train['category'] == 'business'], 2)
         show_ngram(df_train[df_train['category'] == 'business'], 3)
```

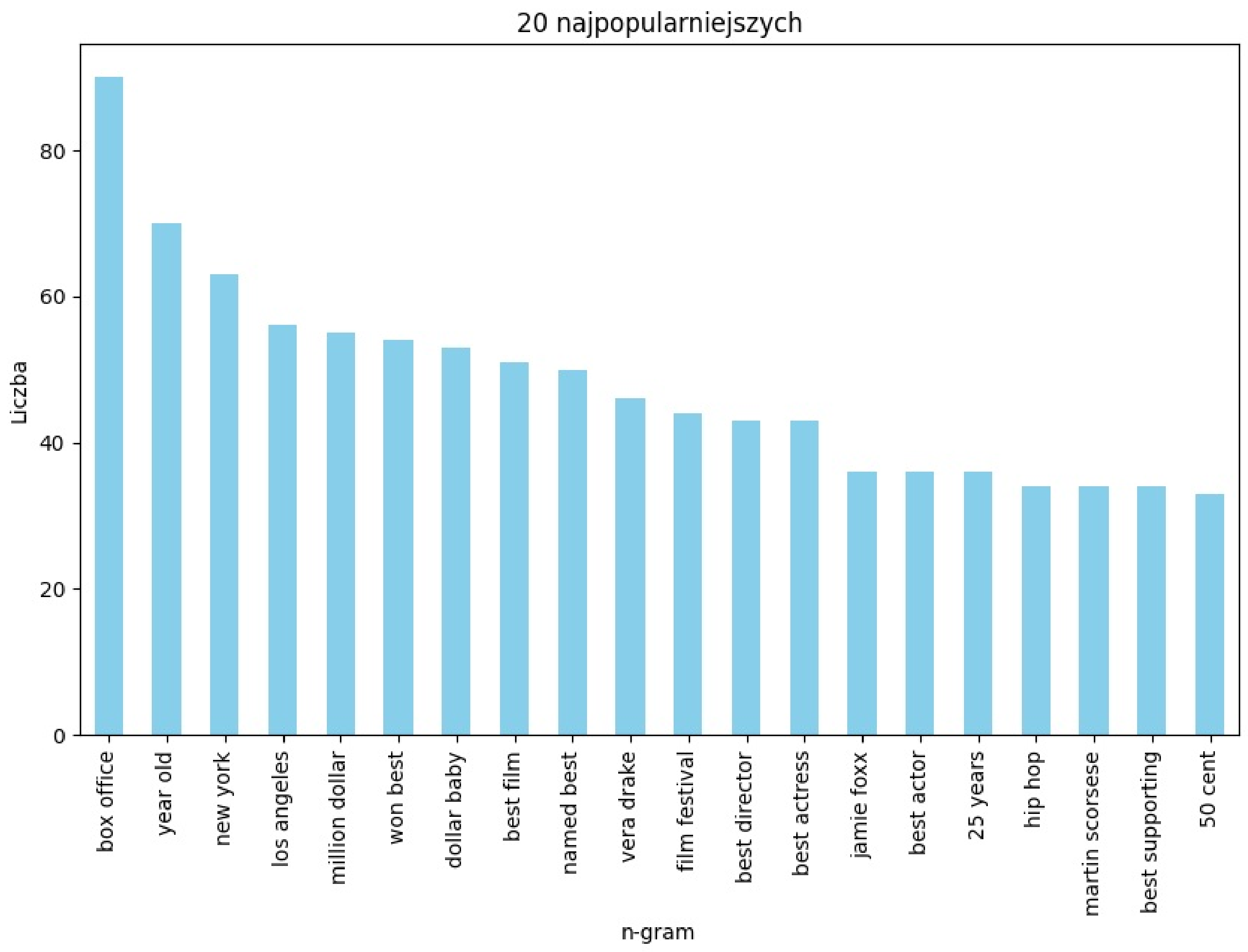


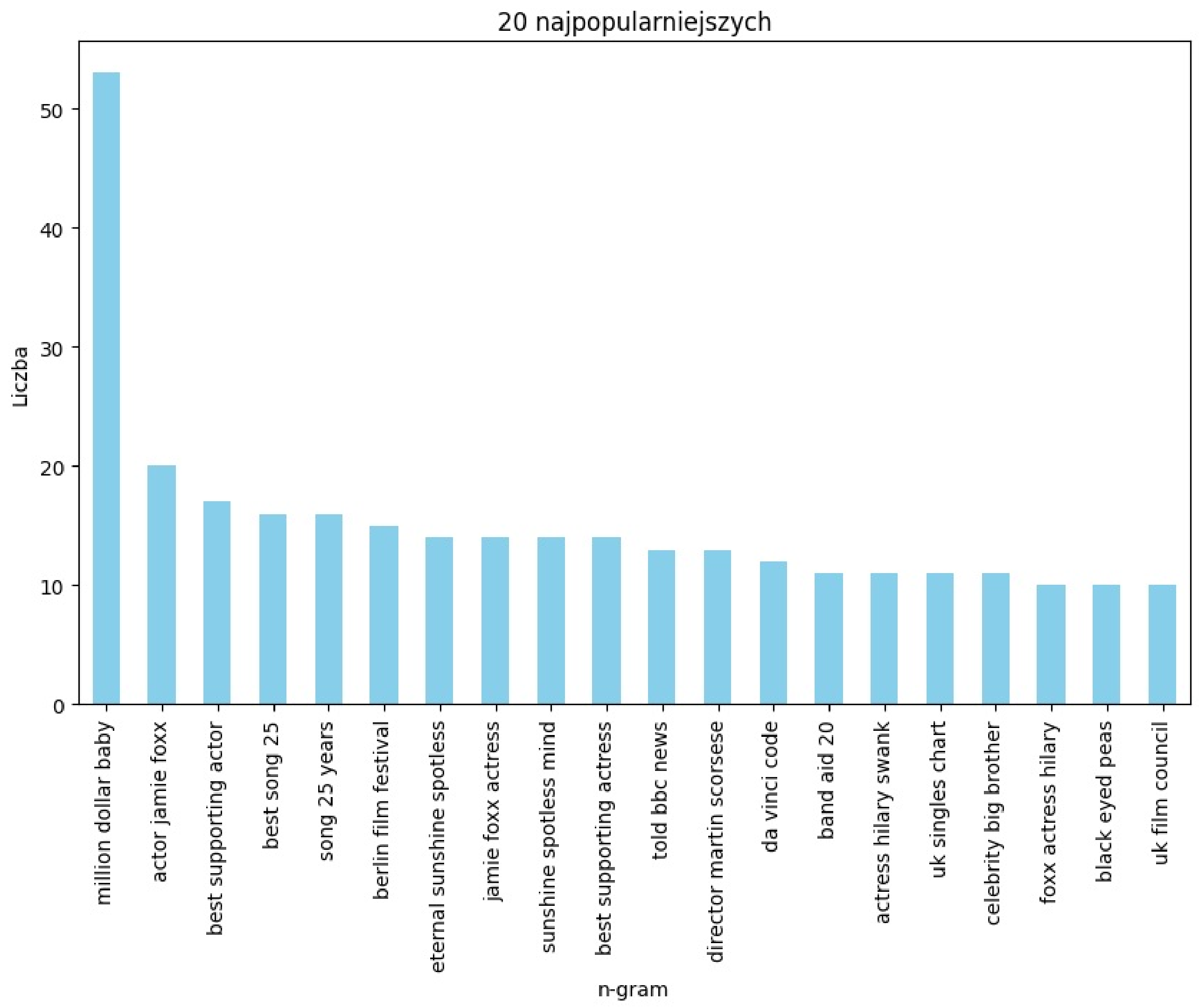


20 najpopularniejszych
Liczba
n-gram
0
5
10
15
20
london stock exchange
securities exchange commission
reuters news agency
gross domestic product
international monetary fund
chief financial officer
high oil prices
current account deficit
president george bush
chapter 11 bankruptcy
told bbc news
fraud tax evasion
world trade organisation
told reuters news
founder mikhail khodorkovsky
pre tax profits
office national statistics
president vladimir putin
political ambitions founder
chairman chief executive

Entertainment

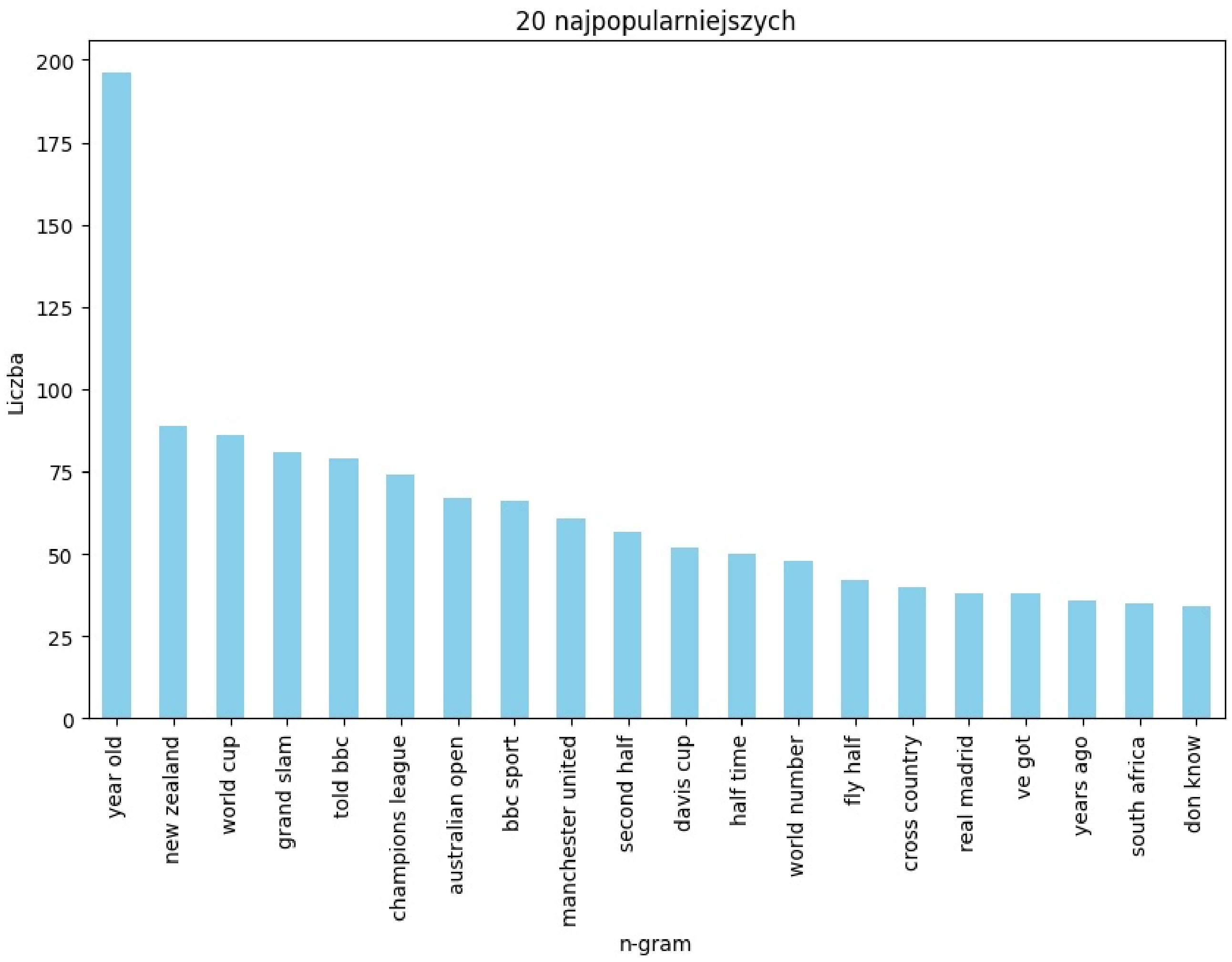
```
In [25]: show_ngram(df_train[df_train['category'] == 'entertainment'], 2)
         show_ngram(df_train[df_train['category'] == 'entertainment'], 3)
```

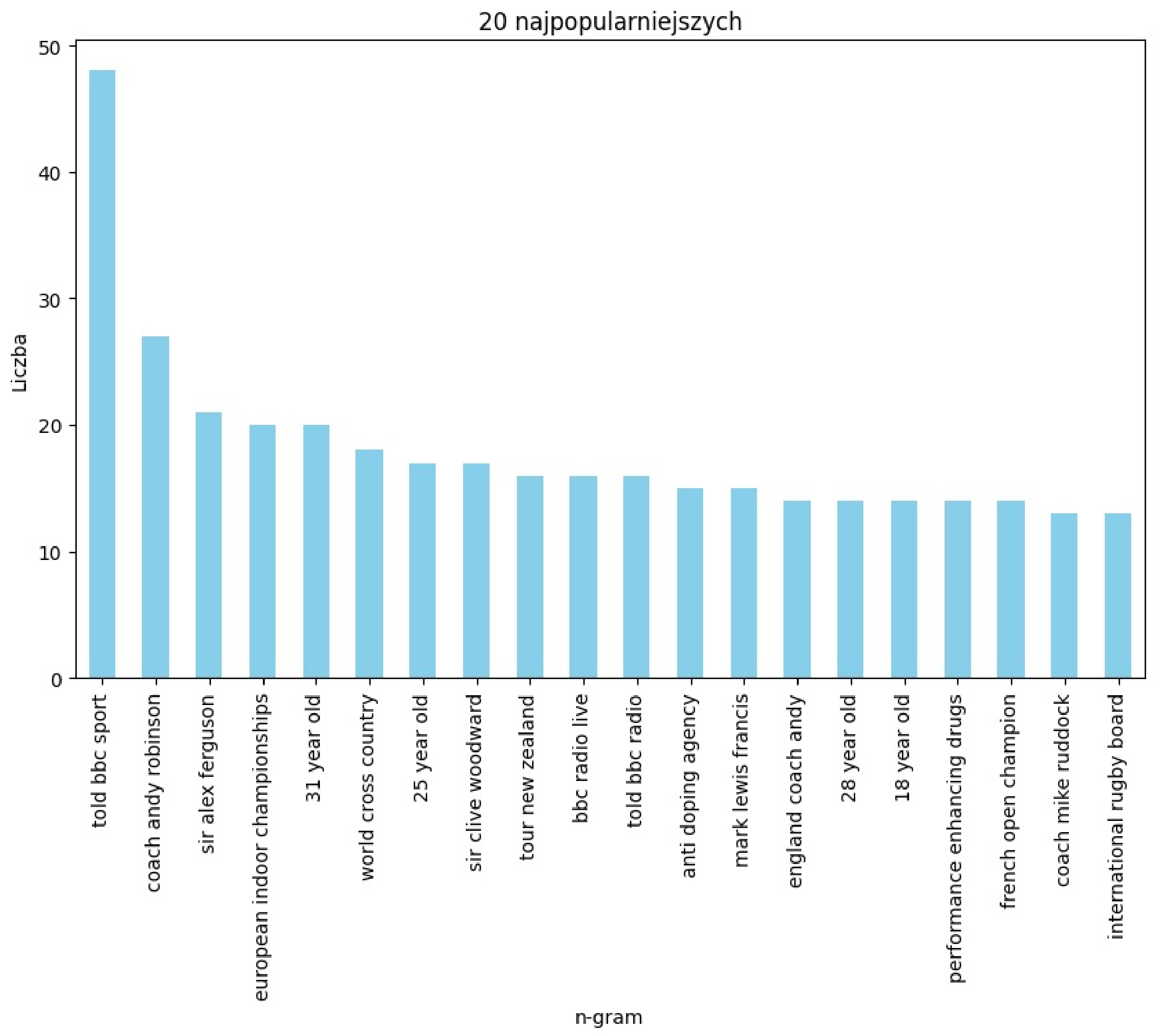



20 najpopularniejszych
Liczba
n-gram
0
10
20
30
40
50
million dollar baby
actor jamie foxx
best supporting actor
best song 25
song 25 years
berlin film festival
eternal sunshine spotless
jamie foxx actress
sunshine spotless mind
best supporting actress
told bbc news
director martin scorsese
da vinci code
band aid 20
actress hilary swank
uk singles chart
celebrity big brother
foxx actress hilary
black eyed peas
uk film council

Sport

In [26]:
```
show_ngram(df_train[df_train['category'] == 'sport'], 2)
show_ngram(df_train[df_train['category'] == 'sport'], 3)
```





Jak działa TensorFlow Hub Universal Sentence Encoder (USE)?

Wykorzystam gotowę narzędzie TensorFlow Hub Universal Sentence Encoder (USE), które przekształca zdania w wektory. Te wektory mogą być używane do porównywania zdań, klasyfikowania tekstu, grupowania i wielu innych zadań. Oto jakie kroki ono wykonuje:

1. Tokenizacja

Na początek, USE dzieli każde zdanie na indywidualne "tokeny", które są zazwyczaj pojedynczymi słowami. Na przykład, zdanie "Kot goni psa" zostanie podzielone na trzy tokeny: "Kot", "goni", "psa".

2. Wektoryzacja

Następnie, USE przekształca każde słowo w wektor. Wektor to po prostu lista liczb, a każde słowo ma swój unikalny wektor. Ten wektor jest wynikiem nauki modelu na dużych ilościach danych i reprezentuje znaczenie słowa, np,

- Kot: [0.1, 0.2, 0.3]
- goni: [0.4, 0.5, 0.6]
- psa: [0.7, 0.8, 0.9]

3. Agregacja

Po przekształceniu słów w wektory, USE agreguje te wektory do jednego wektora reprezentującego całe zdanie. Dzięki temu, mimo że zdanie składa się z wielu słów, USE zwraca jeden wektor dla każdego zdania, np.

- (0.1 [Kot] + 0.4 [goni] + 0.7 [psa]) / 3 = 0.4
- (0.2 [Kot] + 0.5 [goni] + 0.8 [psa]) / 3 = 0.5
- (0.3 [Kot] + 0.6 [goni] + 0.9 [psa]) / 3 = 0.6

Stąd otrzymujemy następujący wektor:

- Kot goni psa: [0.4, 0.5, 0.6]

In [6]:
```
embed = hub.load("https://tfhub.dev/google/universal-sentence-encoder/4")
```

In [7]:
```
embed(['Testuję działanie tego narzędzia']).shape
```

Out[7]:
```
TensorShape([1, 512])
```

In [8]:
```
text_embed = embed(df_train['text'])
text_embed
```

Out[8]:
```
<tf.Tensor: shape=(2225, 512), dtype=float32, numpy=
array([[-0.01426967, -0.05849813, -0.05247907, ..., -0.03674946,
         0.03329156,  0.03961363],
       [-0.05193007, -0.05536706, -0.01138865, ...,  0.04775687,
        -0.02261771, -0.00798134],
       [ 0.02818938, -0.06876058, -0.03309833, ..., -0.01645478,
        -0.06960899, -0.00051798],
       ...,
       [ 0.04089567, -0.05428332,  0.05935666, ..., -0.02636497,
        -0.05927081, -0.0520719 ],
       [ 0.04573027, -0.06175508, -0.0551172 , ...,  0.05862619,
         0.04662686,  0.0098803 ],
       [ 0.0154979 , -0.07087282, -0.06155816, ...,  0.03111984,
        -0.070916  , -0.07013521]], dtype=float32)>
```

Redukujemy wymiary wektorów dla przyszłego ułatwienia wizualizacji.

In [10]:
```
x_2d = TSNE(n_components=2, random_state=0).fit_transform(text_embed)
x_2d
```

Out[10]:
```
array([[-26.654179, -35.735382],
       [-24.533154,  -1.452937],
       [ 51.564045,   4.944287],
       ...,
       [ 12.487085, -11.077203],
       [  2.986454,   8.537201],
       [ 50.126324,  16.51249 ]], dtype=float32)
```

Utworzymy nowy zbiór danych gdzie kolumnami będą wartości powyższego wektora + liczba wszystkich wystąpień danego słowa.

In [11]:
```
df = pd.DataFrame()

x_2d_df = pd.DataFrame(x_2d)
for column in x_2d_df.columns:
    df[column] = x_2d_df[column]

df['category'] = df_train['category']

category_counts = pd.DataFrame(df_train['category'].value_counts()).reset_index().rename(columns={"index":"x",

df = pd.merge(left=df, right=category_counts, on='category', how='left')
df
```

Out[11]:

	0	1	category	count
0	-26.654179	-35.735382	tech	401
1	-24.533154	-1.452937	business	510
2	51.564045	4.944287	sport	511
3	55.048420	18.450281	sport	511
4	6.972558	-50.380852	entertainment	386
...	...	...	...	...
2220	-51.814484	27.367819	business	510
2221	-5.798736	18.739321	politics	417
2222	12.487085	-11.077203	entertainment	386
2223	2.986454	8.537201	politics	417
2224	50.126324	16.512489	sport	511

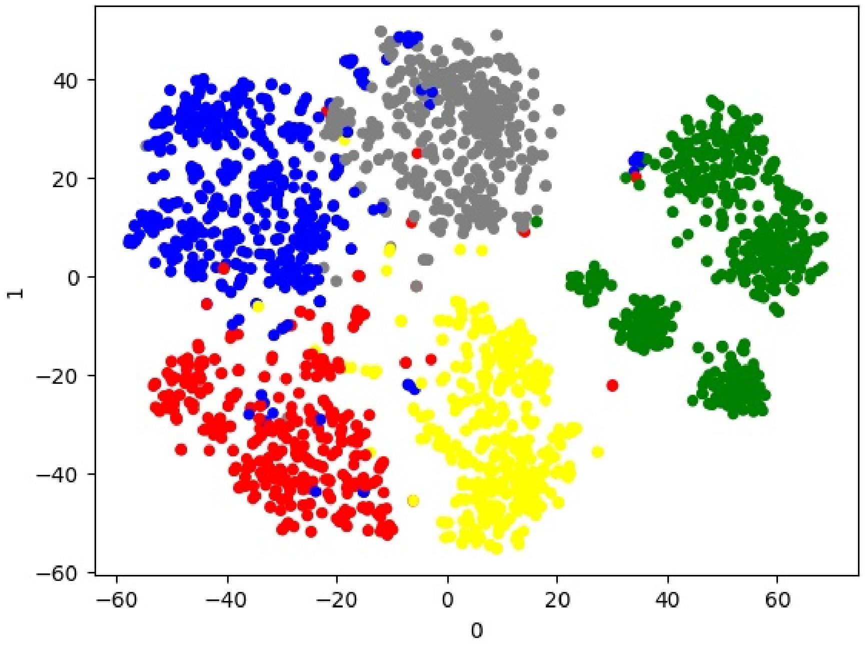
2225 rows × 4 columns

Legenda dla poniższej grafiki:

- *tech* - czerwony
- *business* - niebieski
- *sport* - zielony
- *entertainment* - żółty
- *politics* - szary

In [12]:
```
colormMap = df['category'].map({'tech':'red','business':'blue','sport':'green','entertainment':'yellow','politi
df.plot.scatter(x = 0, y=1, c=colormMap)
```

Out[12]:
```
<Axes: xlabel='0', ylabel='1'>
```



Jak widzimy już na pierwszy rzut oka można zobaczyć bardzo ładnie zrobione grupy danych kategorii (klastery). Oczywiście, niektóry się mieszają, szczególnie Technologia <-> Biznes <-> Polityka.

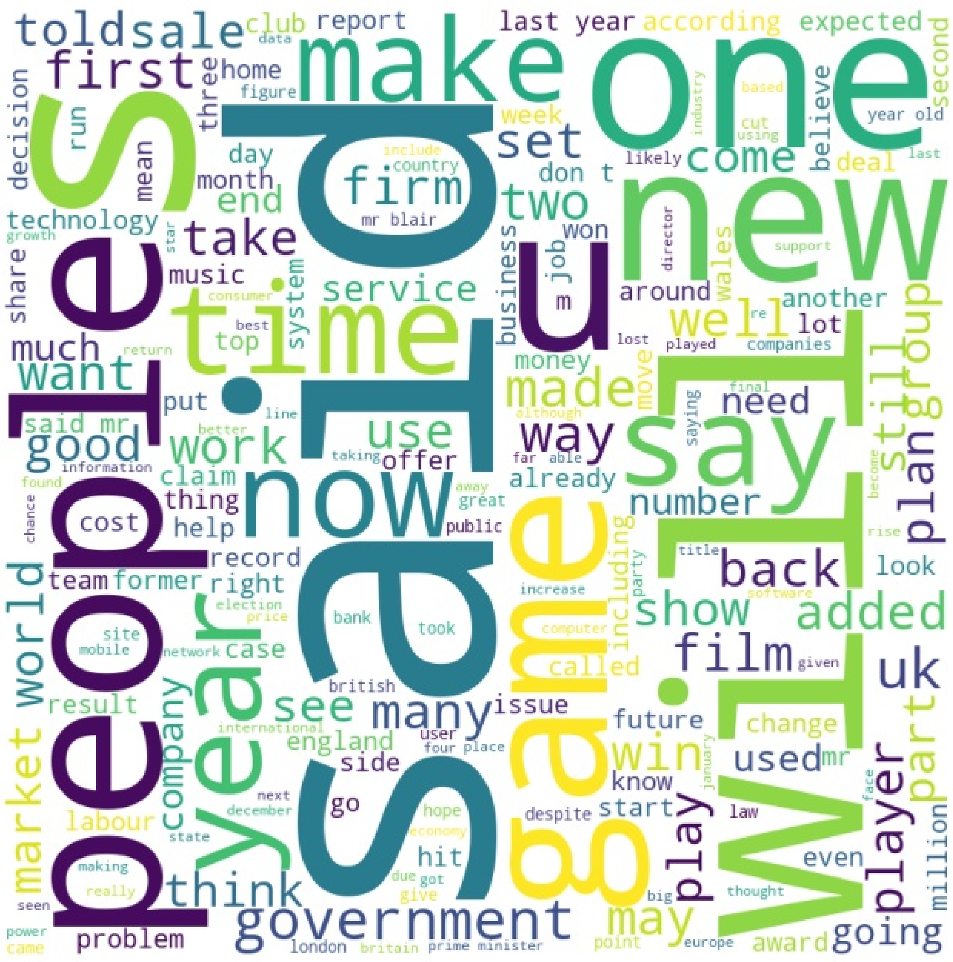
Dosyć ładną rozdzieloną grupę tworzy sport. Można z tego wywnioskować, że pobrane przez nas danę, są dobrej jakości i mogą był łatwo oddzielone od siebie.

Text preprocessing

Zacznijmy od chmury słów. Jest to graficzna reprezentacja występujących słów w podanym tekście.

In [13]:
```
allTextFromColumn = df_train['text']
combinedText = " ".join(text for text in allTextFromColumn)
wordcloud = WordCloud(background_color='white', width = 800, height = 800,).generate(combinedText)
```

In [14]:
```
# Wyświetlamy chmurę słów
plt.figure(figsize = (8, 8))
plt.imshow(wordcloud)
plt.axis("off")
plt.show()
```



Zdefiniujmy teraz przydatne nam funkcje do czyszczenia tekstu. Stworzyłem także funkcję lemmatization, służącą do lematyzacji.

Lematyzacja

Lematyzacja polega na przekształceniu słowa do jego podstawowej formy, inaczej lematem. Dla przykładu, słowa "run", "runs", "ran", i "running" wszystkie mają ten sam lemat, "run". Jest to to samo znaczenie, ale różnią się po prostu formą gramatyczną.

Innym podejściem od lematyacji jest **stemming**, polegający na usuwaniu końcówek słów i zostawiania ich "rdzenia". Przykłady sytuacji, gdzie lematyzacja radzi sobie lepiej od stemmingu:

1. **Słowa o różnych formach, ale o tej samej podstawie:**

Dla słów takich jak "am", "are" i "is", stemowanie może nie dać sensownego wyniku, podczas gdy lematyzacja przekształci te słowa do ich podstawowej formy, czyli "be". 2. **Słowa o różnych znaczeniach w różnych kontekstach:** Rozważmy dwa zdania: 1. I am attending a meeting 2. I am meeting my friend

W stemmingu najprawdopodobniej otrzymamy z obudwu zdań słowo "meet", co przeczy sensowi zdania.

W lematyzacji najprawdopodobniej otrzymamy w pierwszym zdaniu "meeting" wciąż, a w drugim zostanie ono zmniejszone do "meet". Lematyzacja, zachowując kontekst, jest w stanie lepiej utrzymać poprawność gramatyczną i semantykę w zdaniach.

In [18]:
```
def filtered_text(text):
    # usuwamy znaki, które nie są ASCII
    text = ''.join([x for x in text if x in string.printable])
    # usuwamy stopwordy
    text = ' '.join([x for x in text.split() if x not in stopwords.words("english")])
    # lematyzacja o, której jest wyżej
    lm = WordNetLemmatizer()
    text = ' '.join([lm.lemmatize(x, pos='v') for x in text.split()])

    text = text.lower()

    return text
```

Sprawdźmy jak zachowa się ta funkcja do przykładu z lematyzacji wymienionego wyżej.

In [19]:
```
print(filtered_text('I am attending a meeting, I am meeting my friend.'))
```
```
i attend meeting, i meet friend.
```

In [20]:
```
df_train["filtered_text"] = df_train["text"].apply(filtered_text)
```

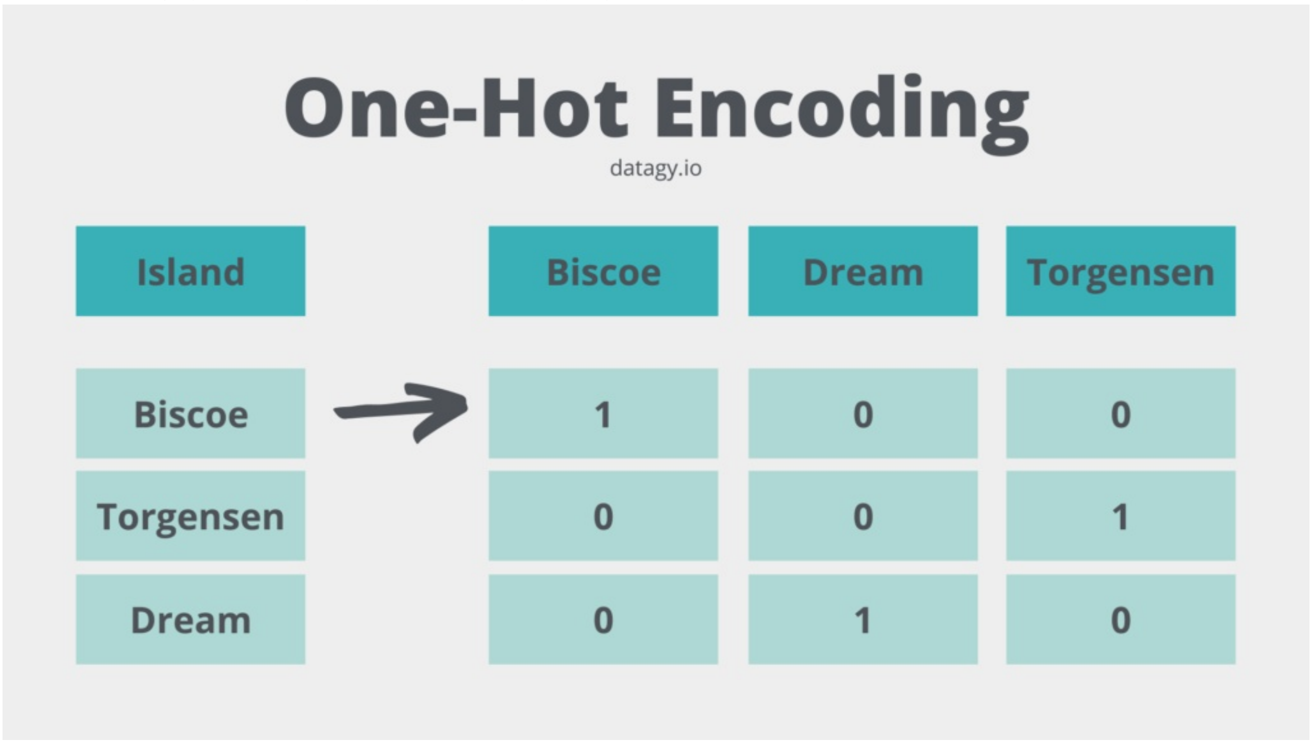
```
In [21]: df_train["filtered_text"]
```

```
Out[21]: 0       tv future hand viewers home theatre systems pl...
         1       worldcom boss leave book alone former worldcom...
         2       tigers wary farrell gamble leicester say rush ...
         3       yeading face newcastle fa cup premiership side...
         4       ocean twelve raid box office ocean twelve crim...
                                     ...
         2220    cars pull us retail figure us retail sales fel...
         2221    kilroy unveil immigration policy ex-chatshow h...
         2222    rem announce new glasgow concert us band rem a...
         2223    political squabble snowball become commonplace...
         2224    souness delight euro progress boss graeme soun...
         Name: filtered_text, Length: 2225, dtype: object
```

```
In [22]: sentences = df_train["filtered_text"].apply(lambda x: x.lower()).tolist()
```

```
In [23]: embed_matrix = []
         for sentence in sentences:
             embed_matrix.append(np.array(embed([sentence])[0]).tolist())
```

Dla ułatwienia algorytmom kategoryzacji przekonwertujmy kategorię z tekstu na binarną zmienną.



```
In [24]: lb = LabelBinarizer().fit(list(set(df_train['category'].tolist())))
         lb.transform(df_train['category'])[:10]
```

```
Out[24]: array([[0, 0, 0, 0, 1],
                [1, 0, 0, 0, 0],
                [0, 0, 0, 1, 0],
                [0, 0, 0, 1, 0],
                [0, 1, 0, 0, 0],
                [0, 0, 1, 0, 0],
                [0, 0, 1, 0, 0],
                [0, 0, 0, 1, 0],
                [0, 0, 0, 1, 0],
                [0, 1, 0, 0, 0]])
```

Dzielimy dane na testowe i trenujące

```
In [25]: X, Y = np.array(embed_matrix), lb.transform(df_train['category'].tolist())
```

```
In [26]: x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=.3, shuffle=True)
```

```
In [27]: x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

```
Out[27]: ((1557, 512), (668, 512), (1557, 5), (668, 5))
```

Model

Skorzystamy następnie z bibloteki **keras**. Jest to narzędzie do budowania i trenowania modeli sieci neuronowych.

Definiujemy warstwa po warstwie kolejne części naszego modelu. Na końcu kompilujemy nasz model wybierając trzy parametry:

1. ***Optimizer*** algorytm do dostosywania parametrów sieci neuronowej by minimalizować funkcję straty. Najpopularniejszym wyborem jest "Adam"
2. ***Loss*** jest to funkcja straty. Oblicza różnicę między prawdziwymi a przewidywanymi etykietami. Naszym celem jest minimalizacja tej wartości. Categorical_crossentropy jest używana w przypadku, gdy etykiety są one-hot encoded (czyli dla każdej klasy mamy osobną kolumnę z wartościami 0 lub 1, w zależności od przynależności do klasy).
3. ***Metrics*** służą do monitorowania procesu uczenia. Nie są potrzebny w trakcie trenowania, ale są obliczane po każdej epoce trenowania. 'Categorical_accuracy' oblicza dokładność klasyfikacji dla problemów wieloklasowych. To jest procent przypadków, w których prawdziwa etykieta klasy jest taka sama jak przewidywana.

In [57]:
```
model = Sequential()
model.add(tf.keras.Input(shape=512))
model.add(Dense(512, activation='relu'))
model.add(Dense(126, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(5, activation='softmax'))
model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics ='categorical_accuracy')
model.build()
```

In [58]:
```
model.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 dense_12 (Dense)            (None, 512)               262656    
                                                                 
 dense_13 (Dense)            (None, 126)               64638     
                                                                 
 dense_14 (Dense)            (None, 64)                8128      
                                                                 
 dense_15 (Dense)            (None, 5)                 325       
                                                                 
=================================================================
Total params: 335,747
Trainable params: 335,747
Non-trainable params: 0
_________________________________________________________________
```

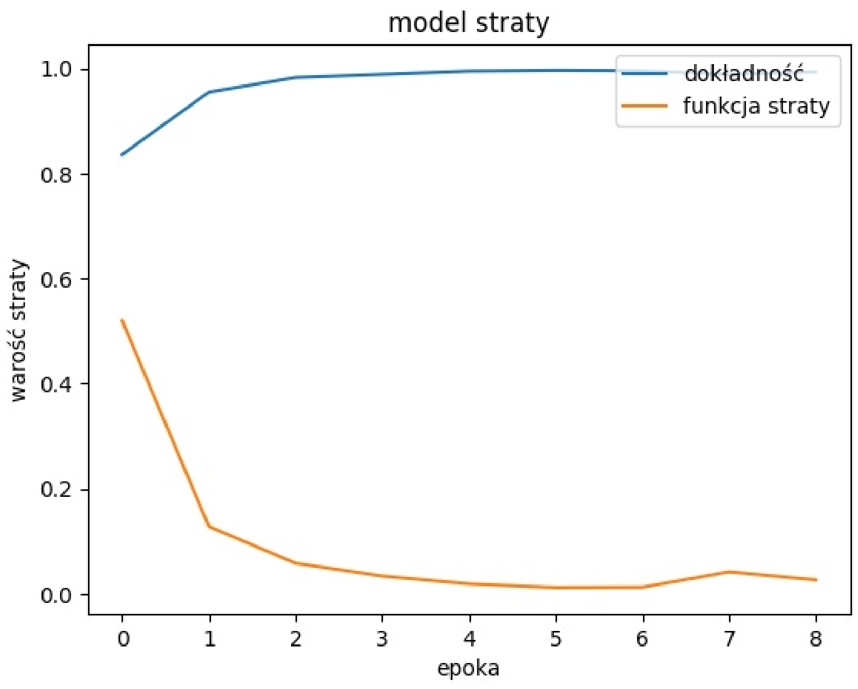
In [59]:
```
es_callback = keras.callbacks.EarlyStopping(monitor='val_loss', patience=5)
```

In [60]:
```
model.fit(x_train, y_train, epochs=100, batch_size=10, validation_split=0.15, callbacks=[es_callback])
```

```
Epoch 1/100
133/133 [==============================] - 1s 4ms/step - loss: 0.5197 - categorical_accuracy: 0.8360 - val_loss:
0.1812 - val_categorical_accuracy: 0.9402
Epoch 2/100
133/133 [==============================] - 0s 3ms/step - loss: 0.1280 - categorical_accuracy: 0.9546 - val_loss:
0.2296 - val_categorical_accuracy: 0.9188
Epoch 3/100
133/133 [==============================] - 0s 3ms/step - loss: 0.0581 - categorical_accuracy: 0.9826 - val_loss:
0.1571 - val_categorical_accuracy: 0.9487
Epoch 4/100
133/133 [==============================] - 0s 3ms/step - loss: 0.0336 - categorical_accuracy: 0.9887 - val_loss:
0.0866 - val_categorical_accuracy: 0.9615
Epoch 5/100
133/133 [==============================] - 0s 3ms/step - loss: 0.0188 - categorical_accuracy: 0.9947 - val_loss:
0.0970 - val_categorical_accuracy: 0.9615
Epoch 6/100
133/133 [==============================] - 0s 3ms/step - loss: 0.0117 - categorical_accuracy: 0.9962 - val_loss:
0.1312 - val_categorical_accuracy: 0.9573
Epoch 7/100
133/133 [==============================] - 0s 3ms/step - loss: 0.0123 - categorical_accuracy: 0.9955 - val_loss:
0.1607 - val_categorical_accuracy: 0.9487
Epoch 8/100
133/133 [==============================] - 0s 3ms/step - loss: 0.0416 - categorical_accuracy: 0.9887 - val_loss:
0.2629 - val_categorical_accuracy: 0.9359
Epoch 9/100
133/133 [==============================] - 0s 3ms/step - loss: 0.0264 - categorical_accuracy: 0.9932 - val_loss:
0.1166 - val_categorical_accuracy: 0.9615
```

Out[60]:
```
<keras.callbacks.History at 0x18a41eaa810>
```

In [61]:
```
history = model.history

plt.plot(history.history['categorical_accuracy'])
plt.ylabel('dokładność')
plt.xlabel('epoka')
plt.plot(history.history['loss'])
plt.title('model straty')
plt.ylabel('warość straty')
plt.xlabel('epoka')
plt.legend(['dokładność', 'funkcja straty'], loc='upper right')
plt.show()
```



Test na kilku artykułach z internetu niepowiązanych z zbiorem uczącym i testowym

Polityka:

- https://edition.cnn.com/2023/05/10/politics/ukraine-russia-putin-trump-town-hall/index.html
- https://edition.cnn.com/2023/05/10/politics/cnn-town-hall-trump/index.html
- https://edition.cnn.com/2023/05/10/politics/missouri-transgender-health-care-sports-ban/index.html

Biznes:

- https://edition.cnn.com/2023/05/11/economy/china-april-cpi-ppi-intl-hnk/index.html
- https://edition.cnn.com/2023/05/11/economy/china-nfra-new-chief-intl-hnk/index.html

Tech:

- https://edition.cnn.com/2023/05/09/tech/airbnb-earnings/index.html
- https://edition.cnn.com/2023/05/10/tech/openai-ceo-congress-testifying/index.html
- https://edition.cnn.com/2023/05/10/tech/twitter-calls-messaging-elon-musk/index.html

In [63]:
```
politics1 = """

Trump says he wants people ‘to stop dying’ but won’t say whether he wants Russia or Ukraine to win war
Trump won't say whether he wants Russia or Ukraine to win war
CNN town hall with former President Donald Trump 7 videos

Former President Donald Trump would not say Wednesday night who he thinks should prevail in Russia’s war agains

“I want everybody to stop dying. They’re dying. Russians and Ukrainians. I want them to stop dying,” Trump said

Trump, who would not say whether he wants Ukraine to successfully deter Russia when pressed by Collins, told th

“I think in terms of getting it settled so we stop killing all these people,” he said.

Trump – asked whether he supports providing US military aid to Ukraine by a Republican voter who will be castin

“We’re giving away so much equipment, we don’t have ammunition for ourselves right now,” he said. “We don’t hav

The White House last week estimated that the Russian military has suffered more than 100,000 casualties since D

White House National Security Council spokesperson John Kirby declined at the time to provide information on Uk

Pressed by Collins on the conflict, Trump replied, “I’ll say this: I want Europe to put up more money.”

The US has provided Ukraine with $36.9 billion in military aid since the beginning of the war in February 2022.
Trump: Putin ‘smart guy’ who ‘made a tremendous mistake’
```

```
While Trump said he would meet with Ukrainian President Volodymyr Zelensky and Russian President Vladimir Putin

The International Criminal Court’s chief prosecutor Karim Khan told CNN in March he believes Putin could stand

The ICC issued arrest warrants in March for Putin and Russian official Maria Lvova-Belova for an alleged scheme

Thousands of Ukrainian children have been subjected to forced deportations by Russia, according to Zelensky, wh

“If you say he’s a war criminal it’s going to be a lot tougher to make a deal to make this thing stopped,” Trum

Trump called Putin “a smart guy,” but said the Russian leader “made a tremendous mistake.”

“Of course he’s smart. They want you to say he’s a stupid person. He’s not a stupid person and he’s very cunnin

When asked to elaborate, Trump said, “His mistake was going in. He would have never gone in if I was president,

CNN’s Tori B. Powell, Elise Hammond, Maureen Chowdhury, Amir Vera, Caitlin Hu, Betsy Klein, DJ Judd, Oren Liebe

"""
```

In [64]:

```
politics2 = """

Trump again refuses to concede 2020 election while taking questions from New Hampshire GOP primary voters
'Outrageous': Legal analyst responds to Trump's comments on E. Jean Carroll trial
CNN town hall with former President Donald Trump 7 videos

Former President Donald Trump, the frontrunner for the GOP presidential nomination in 2024, once again refused

Taking questions from GOP primary voters at the town hall moderated by “CNN This Morning” anchor Kaitlan Collin

The town hall at Saint Anselm College – his first appearance on CNN since 2016 – came as unprecedented legal cl

The audience of Republicans and undeclared voters who plan to vote in the GOP primary cheered Trump throughout

The former president said he would pardon “a large portion” of the rioters at the US Capitol on January 6, 2021

“I am inclined to pardon many of them,” Trump said Wednesday night.

When Collins pressed Trump on the Manhattan federal jury finding Trump sexually abused Carroll in a luxury depa

When asked if the jury’s decision would deter women from voting for him, the former president said, “No, I don’

Trump insulted Carroll, former House Speaker Nancy Pelosi and even Collins when she pressed him on a question a

“It’s very simple – you’re a nasty person, I’ll tell you,” Trump said on stage.

Trump also took questions from New Hampshire voters on the economy and policy issues, such as abortion. The for

Trump suggested Republicans should refuse to raise the debt limit if the White House does not agree to spending

“I say to the Republicans out there – congressmen, senators – if they don’t give you massive cuts, you’re going

When Collins asked him to clarify whether the US should default if the White House doesn’t agree to cuts, Trump
Multiple investigations into Trump

Trump pleaded not guilty last month to 34 felony counts of falsifying business records. Trump also faces potent

Still, the twice-impeached former president has repeatedly said that any charges will not stop him from running

Trump was pressed on the investigation into his handling of classified documents and why he didn’t return all o

The FBI obtained a search warrant and retrieved more than 100 classified documents from Trump’s Florida resort

Asked during the town hall whether he showed the classified documents to anyone at Mar-a-Lago, Trump said, “Not
Trump doesn’t say if he wants Ukraine to win

The former president would not say whether he wants Russia or Ukraine to win the war during Wednesday’s town ha

“I don’t think in terms of winning and losing. I think in terms of getting it settled so we stop killing all th

When asked again whether or not the former president wants Ukraine to win, Trump did not answer directly, but i

“Russians and Ukrainians, I want them to stop dying,” Trump said. “And I’ll have that done in 24 hours.”

Trump said he thinks that “(Russian President Vladimir) Putin made a mistake” by invading Ukraine, but he stopp

That’s something that “should be discussed later,” Trump said.

“If you say he’s a war criminal, it’s going to be a lot tougher to make a deal to make this thing stopped,” he
Trump leads in GOP primary polls

While a handful of rivals have entered the Republican presidential primary – and Trump’s biggest potential riva
```

```
Trump’s participation in the town hall was indicative of a broader campaign strategy to try to expand his appea

There have been warning signs for the GOP that the obsession with the 2020 election isn’t palatable beyond the 

But that didn’t mean Trump was ready to acknowledge the reality that he lost the 2020 election. And if he becom

“If I think it’s an honest election, I would be honored to,” he said.

This story has been updated with additional details from the town hall.

"""
```

In [65]:
```
politics3 = """


Missouri lawmakers passed two bills Wednesday that would bar transgender athletes from playing on school sports

The state House passed both bills largely along party lines Wednesday, after the state Senate approved them in 

One of the bills passed, SB49, would limit minors’ access to gender-affirming care, which spans a range of evid

If enacted, the bill would bar health care providers from performing gender transition surgeries, although such

Puberty blockers and hormone treatments for minors would also be banned until August 2027, but the legislation 

This bill follows an emergency rule issued by the state’s Republican Attorney General Andrew Bailey last month 

Republicans, including the governor, have argued that the bill is about protecting minors from permanent health

“All children, regardless of their gender or orientation, are invaluable and should not be subjected to potenti

However, opponents slammed the lawmakers who voted in favor of the bill, saying that “erasing trans youth is ju

“Gender-affirming care saves lives,” said the American Civil Liberties Union chapter of Missouri in a statement

Lawmakers on Wednesday also passed SB39, which prohibits public and private schools, including colleges, from a

“The presence of biological males in women’s sports limits fair competition for hard-working female athletes,” 

However,  a 2017 report  in the journal Sports Medicine that reviewed several related studies found “no direct 

So far this year, 13 states have placed restrictions on transgender youths’ access to gender-affirming care, in

"""
```

In [66]:
```
bussiness1 = """

China’s inflation cools to 0.1%, its slowest pace in two years
Beijing's economy is rebounding after Covid but the city still has work to do

Deflationary pressure in China is worsening as consumer prices increased at their slowest pace in two years, su

The consumer price index rose by just 0.1% in April from a year ago, the lowest rate of inflation since Februar

The producer price index, which measures factory-gate prices, declined by 3.6%, marking the biggest contraction

Prices are stagnating or falling in the country despite the fact that the People’s Bank of China (PBOC), the ce

There have been some signs of recovery in demand after the end of pandemic restrictions late last year, but con

“The subdued inflation readings suggest post-Covid recovery momentum continued to weaken in April,” Nomura anal

The weak property sector recovery likely has exerted “persistent” downward pressure on the factory-gate prices,

Real estate contributes as much as 30% to China’s GDP. It is still in the midst of a historic downturn, with ne

A slump in the property sector affects demand for key raw materials such as steel and cement, which are key par

According to data released Tuesday by the customs authority, imports plunged 7.9% in April, indicating weak dom
A customer browses through discount clothing displayed outside a shop in Beijing on May 10, 2023.

Instead of spending money, people are hoarding cash at a record rate. In the first quarter, deposits held by ho

Looking ahead, Nomura analysts expect consumer inflation to remain “fairly soft” in May, likely around 0.3%. Pr

Although some government economists have recently warned the economy is facing deflationary pressure, Chinese a

A spokesperson from the NBS said late last month the Chinese economy “does not appear deflated” and that deflat

Deflation is bad for the economy because, in such an environment, consumers and companies may put off spending 

Zou Lan, an official with the PBOC, said at a press conference on April 20 that price increases are likely to r
```

```
"""
```

In [67]:
```
bussiness2 = """

China names head of powerful new financial regulator as industry faces greater scrutiny
China banking scandal is affecting thousands of people

China has appointed the head of its powerful new financial watchdog, which was created as part of sweeping refo

Li Yunze, a career banker working for decades at China’s state-owned lenders, will become the Communist Party cl

The NFRA was formed in March when Chinese leader Xi Jinping unveiled the biggest overhaul of the government in (

The regulator is set to replace the existing China Banking and Insurance Regulatory Commission, which supervises
Li Yunze, the newly appointed Communist Party chief of China's National Financial Regulatory Administration. The

Li, 52, is currently the deputy governor of Sichuan, a southwestern province home to 84 million people.

A graduate of Tianjin University, Li worked at China Construction Bank, the second largest lender by assets, fo

In 2016, he left the bank and became vice president of the Industrial and Commercial Bank of China, the country

It’s not unusual for Communist Party officials to be transferred to different roles at various state-owned orga

Currently, several provincial leaders had previous careers in the financial industry, including Wu Qing, vice ma

China’s sprawling financial industry is coming under closer scrutiny as Xi and his key allies have asserted grea

For years, Xi has said the financial industry should better serve the real economy, including making money avai

The sweeping government reform in March reflected his belief that China needs to build a more centralized gover

To further consolidate control, according to analysts, the top anti-graft body has carried out a sweeping anti-

"""
```

In [68]:
```
tech1 = """
irbnb sees record bookings despite recession fears
CNN tried an AI flirt app. It was shockingly pervy

Shares of Airbnb fell more than 10% in trading Wednesday after the travel company offered a more muted outlook

Airbnb on Tuesday reported strong revenue growth and a new record for bookings during the first three months of

The company reported revenue for the quarter grew 20% to $1.8 billion, just beating Wall Street’s estimates. The

“More guests are traveling on Airbnb than ever before,” CEO Brian Chesky said on a call with analysts Tuesday.

“During the quarter, we also saw guests booking trips further in advance, supporting a strong backlog for q2,”

Airbnb also reported net income of some $117 million, compared to a net loss in the same period last year.

But the company’s guidance for the current quarter “disappointed on both the top and bottom line,” analysts at

In its letter to shareholders, Airbnb said it has pulled forward the timing of its marketing spend to the first

Tom White, a senior research analyst at D.A. Davidson, said the lackluster guidance for the current quarter was

Some skeptics of the company “may point to this as evidence that [Airbnb] is struggling to maintain its growth

But White wrote that he believes there continue to be “promising incremental” ways for Airbnb to unlock growth

"""
```

In [69]:
```
tech2= """
 OpenAI CEO Sam Altman will testify before Congress next Tuesday as lawmakers increasingly scrutinize the risks

During Tuesday’s hearing, lawmakers will question Altman for the first time since OpenAI’s chatbot, ChatGPT, to

The groundbreaking generative AI tool has led to a wave of new investment in AI, prompting a scramble among US

Also testifying Tuesday will be Christina Montgomery, IBM’s vice president and chief privacy and trust officer,

“Artificial intelligence urgently needs rules and safeguards to address its immense promise and pitfalls,” said

He added: “I look forward to working with my colleagues as we explore sensible standards and principles to help
"""
```

In [70]:
```
tech3 = """

Twitter is adding calls and encrypted messaging
Tucker Carlson announces he's relaunching his show on Twitter
```

```
Twitter is adding encrypted messaging to the platform Wednesday, and calls will follow shortly, CEO Elon Musk tw

“Release of encrypted DMs [direct messages] V1.0 should happen tomorrow. This will grow in sophistication rapidl

“Coming soon will be voice and video chat from your handle to anyone on this platform, so you can talk to people

The move comes as Musk, who took control of Twitter six months ago, looks for ways to return the platform to gro

Adding calls and encrypted messaging could allow Twitter to compete with Mark Zuckerberg’s Meta, which owns Face

Since taking the company private in October, Musk has turned Twitter on its head. A number of users, celebrities

Right-wing TV host Tucker Carlson said Tuesday he would relaunch his program on Twitter, which he praised as the

"""
```

In [71]:
```
tech1 = filtered_text(tech1)
tech2 = filtered_text(tech2)
tech3 = filtered_text(tech3)
```

In [72]:
```
print(sorted(list(set(df_train['category'].tolist())))[model.predict(embed([tech1])).argmax(axis=1)[0]])
print(sorted(list(set(df_train['category'].tolist())))[model.predict(embed([tech2])).argmax(axis=1)[0]])
print(sorted(list(set(df_train['category'].tolist())))[model.predict(embed([tech3])).argmax(axis=1)[0]])
```

```
1/1 [==============================] - 0s 15ms/step
business
1/1 [==============================] - 0s 14ms/step
tech
1/1 [==============================] - 0s 14ms/step
tech
```

In [73]:
```
bussiness1 = filtered_text(bussiness1)
bussiness2 = filtered_text(bussiness2)
print(sorted(list(set(df_train['category'].tolist())))[model.predict(embed([bussiness1])).argmax(axis=1)[0]])
print(sorted(list(set(df_train['category'].tolist())))[model.predict(embed([bussiness2])).argmax(axis=1)[0]])
```

```
1/1 [==============================] - 0s 15ms/step
business
1/1 [==============================] - 0s 14ms/step
business
```

In [74]:
```
politics1 = filtered_text(politics1)
politics2 = filtered_text(politics2)
politics3 = filtered_text(politics3)
print(sorted(list(set(df_train['category'].tolist())))[model.predict(embed([politics1])).argmax(axis=1)[0]])
print(sorted(list(set(df_train['category'].tolist())))[model.predict(embed([politics2])).argmax(axis=1)[0]])
print(sorted(list(set(df_train['category'].tolist())))[model.predict(embed([politics3])).argmax(axis=1)[0]])
```

```
1/1 [==============================] - 0s 15ms/step
politics
1/1 [==============================] - 0s 14ms/step
politics
1/1 [==============================] - 0s 14ms/step
politics
```

Wyniki

Jak widać, najgorzej poradził sobie z artykułami typu tech, z innymi poradził sobie dobrze. Jest to możliwe z kilku przyczyn, m.in.:

1. Dane są z roku 2004-2005 a ponad 15 lat czasu wystarcza na gigantyczną zmianę terminologi technicznej używanej w mediach.
2. Artykuły technologiczne na stronie cnn nie są czysto technologiczne, lecz poruszają dodatkowe tematy.
    - Artykuł uznany za biznes mówi o zarobkach airbnb, więc tematyka biznesowa

Na zbiorze uczącym i testowym otrzymaliśmy bardzo wysokie wyniki, jednak warto pamiętać, że korzystaliśmy z zewnętrzego wytrenowanego narzędzia.

Bibliografia

1. http://mlg.ucd.ie/datasets/bbc.html
2. https://stackoverflow.com/questions/62700996/tpu-with-tensorflow-v1
3. https://www.tensorflow.org/hub/tutorials/semantic_similarity_with_tf_hub_universal_encoder?hl=pl
4. https://aclanthology.org/D18-2029/
5. https://en.wikipedia.org/wiki/Bigram
6. https://edition.cnn.com/2023/05/10/politics/ukraine-russia-putin-trump-town-hall/index.html
7. https://edition.cnn.com/2023/05/10/politics/cnn-town-hall-trump/index.html
8. https://edition.cnn.com/2023/05/10/politics/missouri-transgender-health-care-sports-ban/index.html

9. https://edition.cnn.com/2023/05/11/economy/china-april-cpi-ppi-intl-hnk/index.html
10. https://edition.cnn.com/2023/05/11/economy/china-nfra-new-chief-intl-hnk/index.html
11. https://edition.cnn.com/2023/05/09/tech/airbnb-earnings/index.html
12. https://edition.cnn.com/2023/05/10/tech/openai-ceo-congress-testifying/index.html
13. https://edition.cnn.com/2023/05/10/tech/twitter-calls-messaging-elon-musk/index.html

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js