In [1]:
```python
import string
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.feature_extraction.text import CountVectorizer
from wordcloud import WordCloud
```

In [2]:
```python
df = pd.read_csv('emotions_preprocessed.csv')
```

Exploratory Data Analysis

In [3]:
```python
df.head()
```

Out[3]:

	text	labels
0	My favourite food is anything I didn't have to...	1
1	Now if he does off himself, everyone will thin...	1
2	WHY THE FUCK IS BAYLESS ISOING	12
3	To make her feel threatened	7
4	Dirty Southern Wankers	12

In [4]:
```python
df.describe()
```

Out[4]:

	labels
count	53994.000000
mean	6.103660
std	4.573331
min	1.000000
25%	1.000000
50%	6.000000
75%	10.000000
max	14.000000

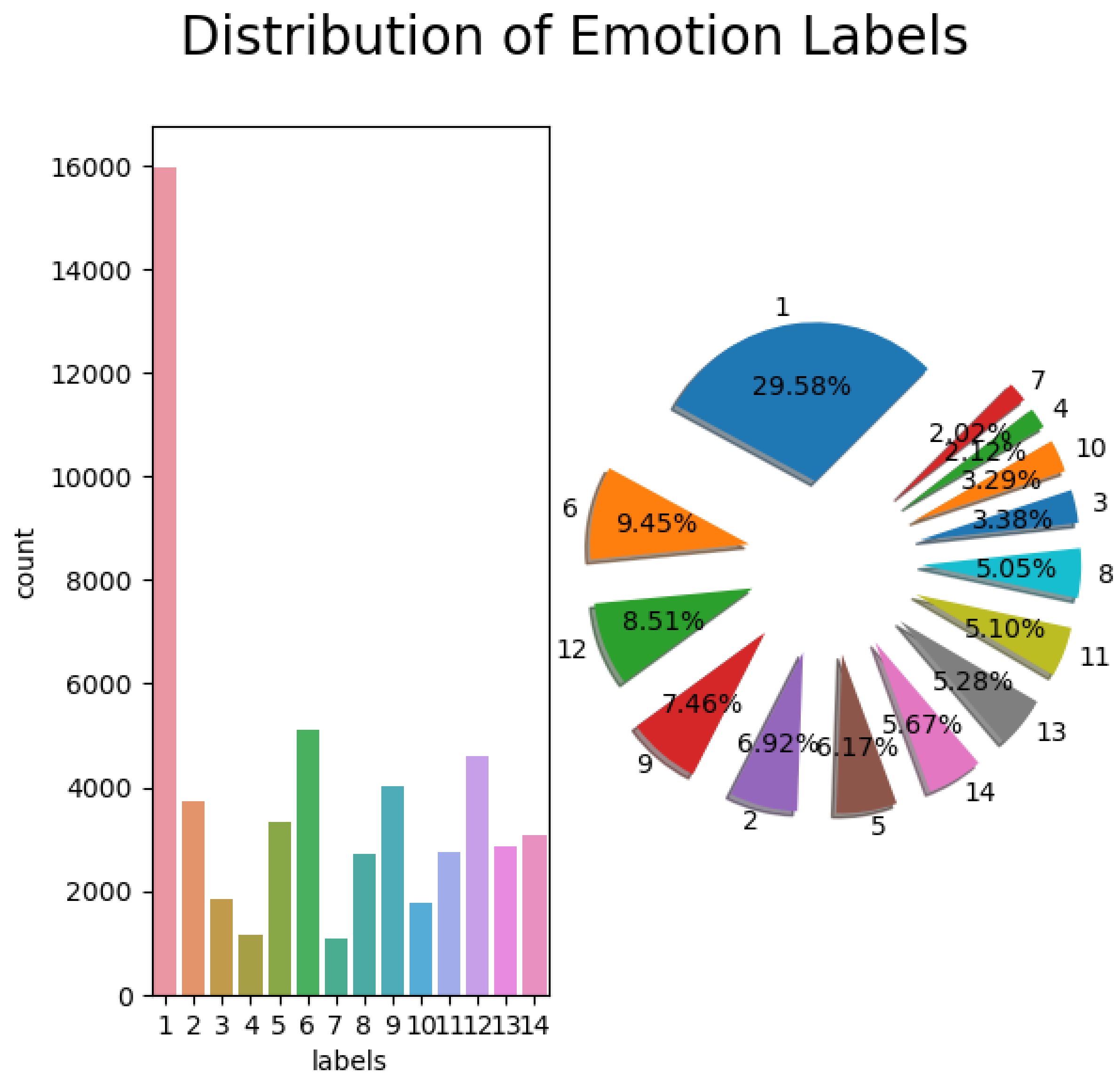
In [5]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53994 entries, 0 to 53993
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    53994 non-null  object
 1   labels  53994 non-null  int64 
dtypes: int64(1), object(1)
memory usage: 843.8+ KB
```

In [6]:
```python
label_counts = df['labels'].value_counts()
label_list = list(label_counts.index.astype(str))
```

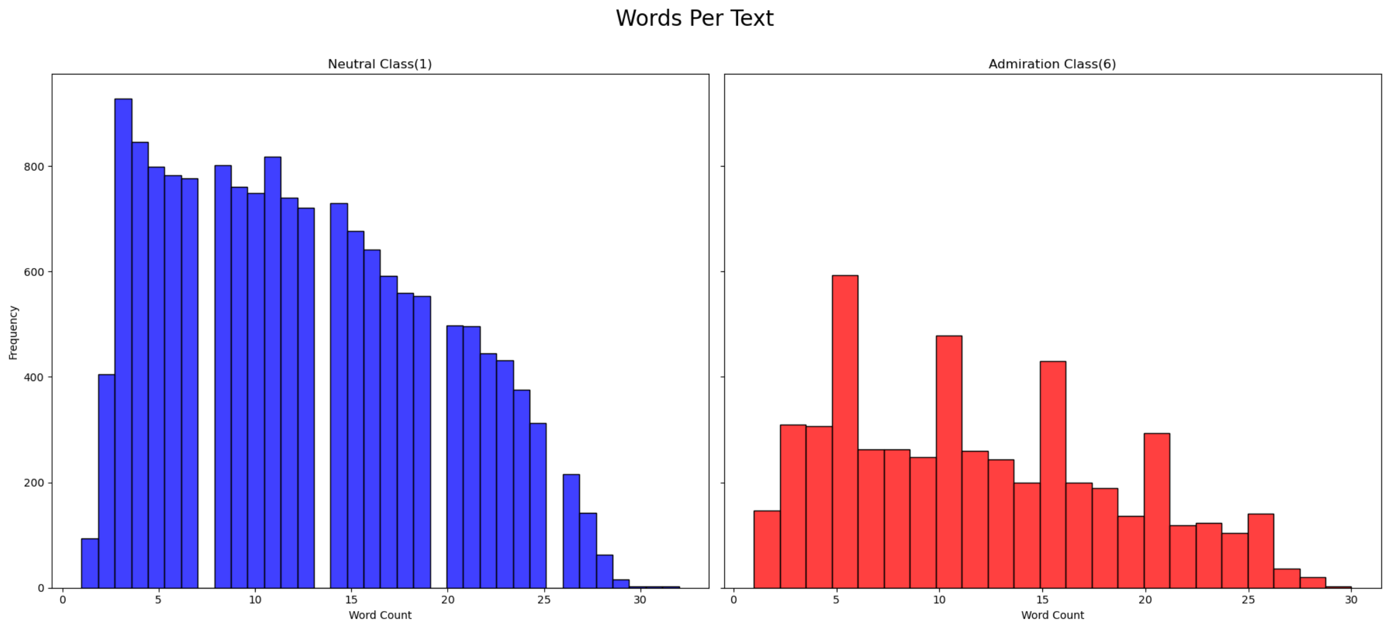
```
fig, axes = plt.subplots(ncols=2, nrows=1, figsize=(6,6), dpi=100)
sns.countplot(df['labels'], ax=axes[0])
axes[1].pie(df['labels'].value_counts(),
            labels=label_list,
            autopct='%1.2f%%',
            shadow=True,
            explode=(0.5,0.5, 0.5,0.6,0.6, 0.6,0.6,0.6, 0.6,0.6,0.6, 0.6,0.6,0.6),
            startangle=45)
fig.suptitle('Distribution of Emotion Labels', fontsize=20)
plt.show()
```

```
/Users/anthonyawobasivwe/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.p
y:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12,
the only valid positional argument will be `data`, and passing other arguments without a
n explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



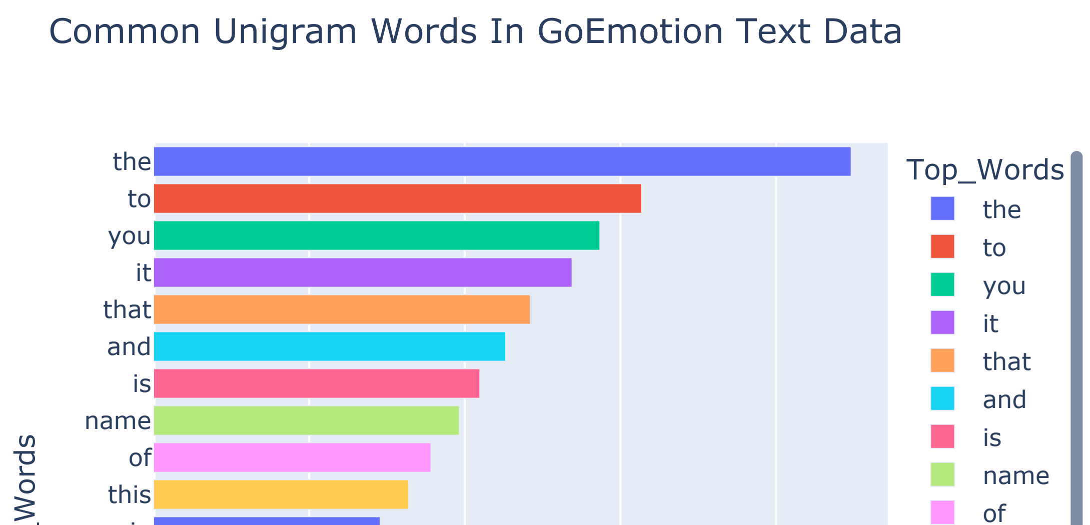
In [7]:
```
#histogram for number of words in text
def plot_word_number_histogram(neutral, admiration):


    fig, axes = plt.subplots(ncols=2, nrows=1, figsize=(18, 8), sharey=True)
    sns.histplot(neutral.str.split().map(lambda x: len(x)), ax=axes[0], color='blue')
    sns.histplot(admiration.str.split().map(lambda x: len(x)), ax=axes[1], color='red')

    axes[0].set_xlabel('Word Count')
    axes[0].set_ylabel('Frequency')
    axes[0].set_title('Neutral Class(1)')
    axes[1].set_xlabel('Word Count')
    axes[1].set_title('Admiration Class(6)')

    fig.suptitle('Words Per Text', fontsize=20, va='baseline')
    fig.tight_layout()
```

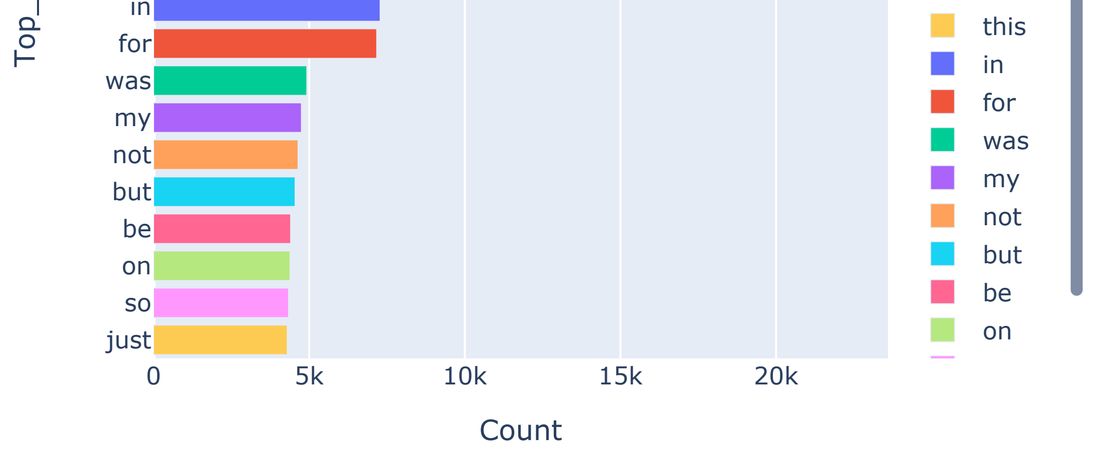
In [8]:
```
#number of words for two most common labels
plot_word_number_histogram(df[df['labels'] == 1]['text'],
                           df[df['labels'] == 6]['text'])
```



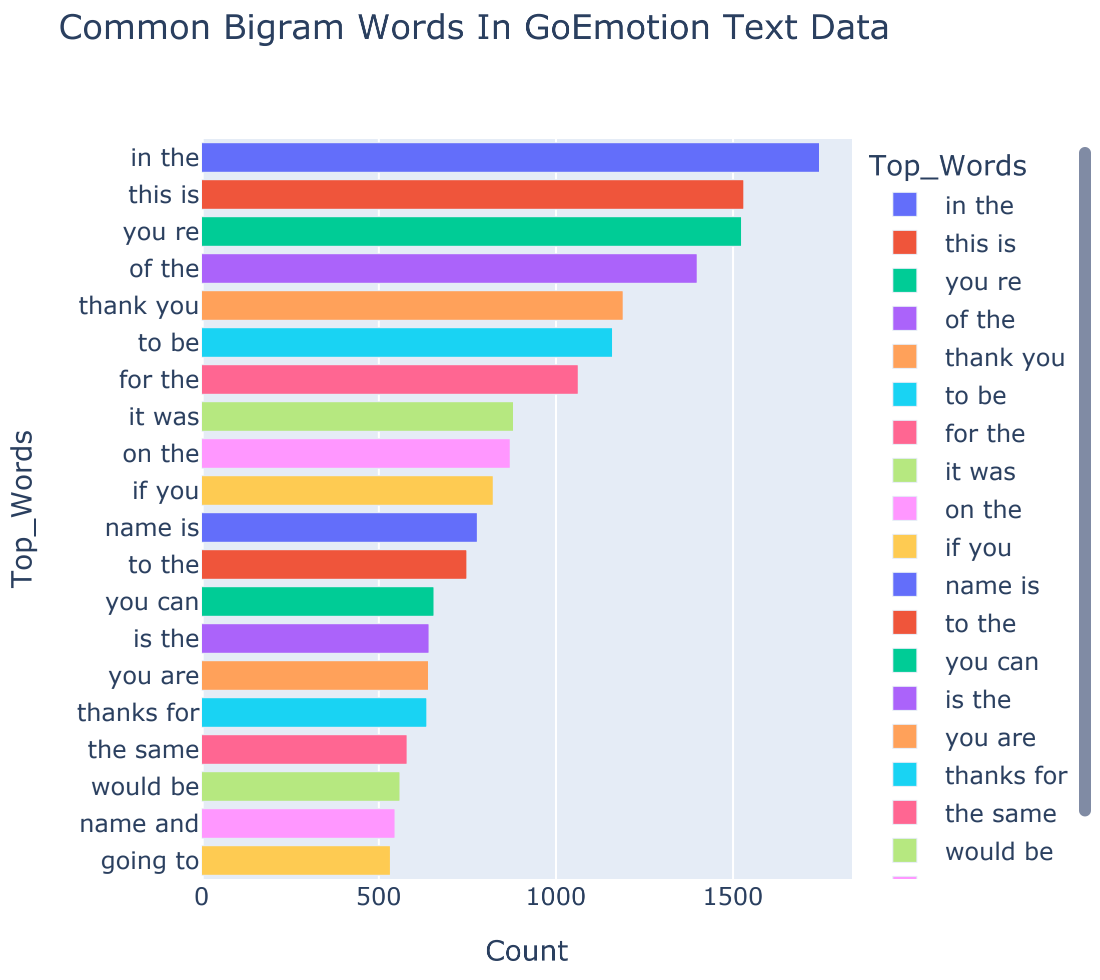
In [9]:
```
def top_ngrams(corpus, num, gram):
    cv = CountVectorizer(ngram_range=(gram, gram)).fit(corpus)
    bow = cv.transform(corpus)
    wordsum = bow.sum(axis=0)
    freq = [(word, wordsum[0, idx]) for word, idx in cv.vocabulary_.items()]
    freq =sorted(freq, key = lambda x: x[1], reverse=True)
    return freq[:num]
```

In [10]:
```
unigram_common = top_ngrams(df.text,20,1)
unigram_common = dict(unigram_common)
unigram_common_df = pd.DataFrame(columns = ["Top_Words" , 'Count'])
unigram_common_df["Top_Words"] = list(unigram_common.keys())
unigram_common_df["Count"] = list(unigram_common.values())
fig = px.bar(unigram_common_df, x="Count", y="Top_Words", title='Common Unigram Words In
             width=550, height=550,color='Top_Words')
fig.show()
```



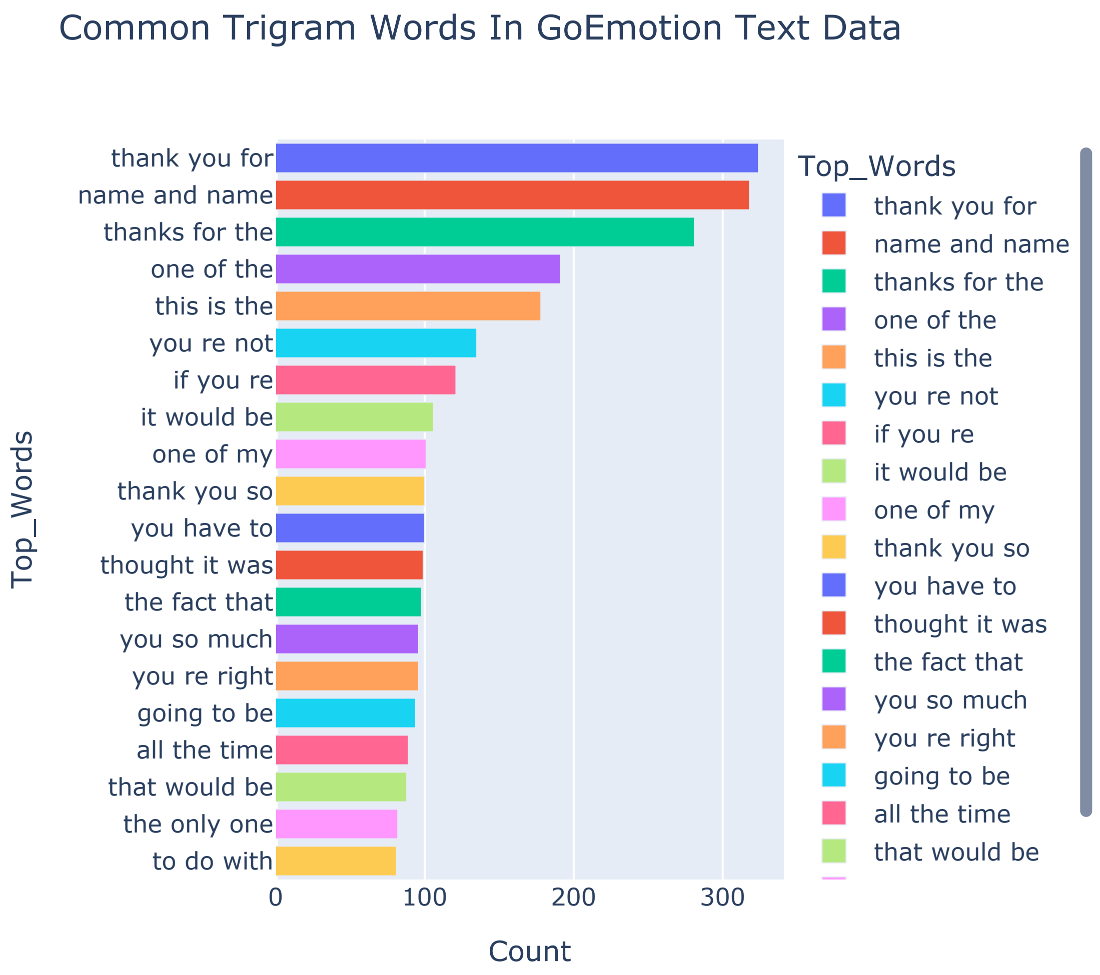


```
In [11]: bigram_common = top_ngrams(df.text,20,2)
         bigram_common = dict(bigram_common)
         bigram_common_df = pd.DataFrame(columns = ["Top_Words" , 'Count'])
         bigram_common_df["Top_Words"] = list(bigram_common.keys())
         bigram_common_df["Count"] = list(bigram_common.values())
         fig = px.bar(bigram_common_df, x="Count", y="Top_Words", title='Common Bigram Words In G
                      width=550, height=550,color='Top_Words')
         fig.show()
```



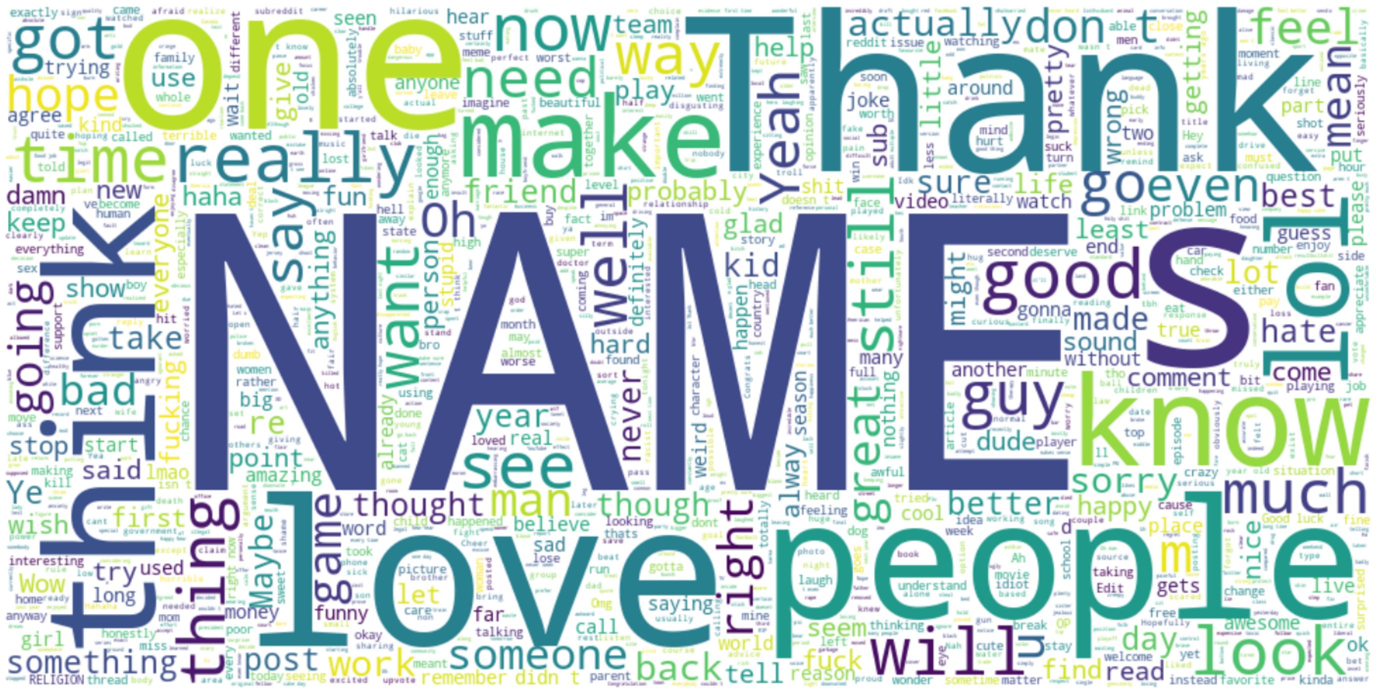
```
In [12]: trigram_common = top_ngrams(df.text,20,3)
         trigram_common = dict(trigram_common)
         trigram_common_df = pd.DataFrame(columns = ["Top_Words" , 'Count'])
```

```
trigram_common_df["Top_Words"] = list(trigram_common.keys())
trigram_common_df["Count"] = list(trigram_common.values())
fig = px.bar(trigram_common_df, x="Count", y="Top_Words", title='Common Trigram Words In
             width=550, height=550,color='Top_Words')
fig.show()
```



In [13]:
```
wcloud = df['text']
wcloud_str = ' '.join(wcloud)
plt.figure(figsize=(18,18))
wc = WordCloud(max_words=1500,width = 1000, height=500,background_color= "white").genera
plt.imshow(wc,interpolation='bilinear')
plt.axis('off')
plt.title("Word count GoEmotions Text",fontsize=20)
plt.show()
```

Word count GoEmotions Text



```
In [ ]: 
```

```
In [ ]: 
```