Ceasefires as bargaining instruments in intrastate conflicts: ceasefire objectives and their effects on peace negotiations

Sticher, V.

Appendix Chapter Five

This appendix contains code that was used for the empirical analysis in chapter five (*War of narratives*). This includes code to download and select conflict and peace process related tweets by presidents Santos and Uribe, code to compile different sets of corpora, and code to conduct the natural language processing analysis. Chapter five details how this code was adapted for the robustness tests.

Scripts containing the code, setup requirements and instructions on how to run the analysis can be found on *Github* at https://github.com/ValerieSt/framing.

Requirements to run the code include Python3 and the following python packages: pandas, spacy, matplotlib, nltk, google.cloud and wordcloud. A Google Cloud account is needed, and authentication credentials need to be copied into the shell file as indicated below.

Note that the results may change over time, as natural language processing libraries and models evolve. The findings, as presented in this thesis, were obtained with the scripts run on 11 December 2019.

Shell file

0-run_scripts.sh

```
#!/bin/(shell)
export GOOGLE_APPLICATION_CREDENTIALS=[PATH][115]
GetOldTweets3 --username "juanmansantos" --since 2012-08-27 --until 2016-10-03 --maxtweets 0
python3 1-select_tweets.py
python3 2-translate.py
python3 3-complete_translation.py
python3 3-complete_translation.py
python3 4a-rename.py
GetOldTweets3 --username "AlvaroUribeVel" --since 2012-08-27 --until 2016-10-03 --maxtweets 0
python3 1-select_tweets.py
python3 2-translate.py
python3 3-complete_translation.py
```

[115] To run the shell, the string [PATH] needs to be replaced with the path to the JSON file that contains the service account key (to authenticate the Google Cloud account).

```
python3 3-complete_translation.py
python3 4b-rename.py
python3 5-create_corpora_entire_period.py
python3 6-create_corpora_fighting_vs_ceasefire.py
python3 7-identifying_past_tense.py
python3 8-identifying_future_tense.py
python3 9-stops_and_lemma.py
python3 10-word_frequencies-wordcloud.py
python3 11-create_order.py
```

Scripts

1-select_tweets.py

```
import pandas as pd
import re
import string
df1 = pd.read_csv("output_got.csv")
fout = open('stats.txt', 'w')
total_rows = df1['text'].count()
fout.write('Total number of tweets: ')
fout.write(str(total_rows))
fout.write('\n')
df2 = pd.DataFrame(columns=['timedate', 'date', 'username', 'text', 'permalink'])
def remove_hashtags(text):
    for i in range(0, 5):
        atpos = text.find('#')
        sppos = text.find(' ',atpos)
        hashtag = text[atpos:sppos]
        text = re.sub(str(hashtag), "", text)
    return text
def remove_mentions(text):
    for i in range(0, 5):
        atpos = text.find('@')
        sppos = text.find(' ',atpos)
        mention = text[atpos:sppos]
        text = re.sub(str(mention), "", text)
    return text
print('Selecting tweets that are likely related to the peace process & removing hashtags, mentions and URLs...')
count = 0
farccount = 0
colombiacount = 0
for i in range(0, total_rows):
```

```
    tweet = df1.text[i]
    tweet = tweet.rstrip()
    tweet = tweet.casefold()
    tweet = re.sub(r"http\S+", "", tweet)
    tweet = re.sub(r"pic.twitter\S+", "", tweet)
    if (tweet.find('farc') is not -1):
        farccount = farccount + 1
    if (tweet.find('farc') == -1 and tweet.find('paz') == -1 and tweet.find('guerra') == -1 and tweet.find('plebiscito') == -1 and
tweet.find('decorazónvotono') == -1 and tweet.find('eldomingovotono') == -1 and tweet.find('todosavotarno') == -1 and
tweet.find('eshoradevotarno') == -1 and tweet.find('votonoycorrijoacuerdos') == -1 and tweet.find('colombiadecideno') == -1
and tweet.find('hagahistoriavoteno') == -1 and tweet.find('conargumentosdigono') == -1 and
tweet.find('encartagenadecimosno') == -1 and tweet.find('cartagenapitano') == -1 and tweet.find('colombiavotano') == -1 and
tweet.find('mirazonparavotarno') == -1 and tweet.find('colombiaconelno') == -1 and tweet.find('todosporelno') == -1 and
tweet.find('unidosporelno') == -1 and tweet.find('decimosnoalosacuerdos') == -1 and tweet.find('colombiasabedecirno') == -
1 and tweet.find('digonoalosacuerdos') == -1 and tweet.find('yonometragoestesapo') == -1 and
tweet.find('conozcalosacuerdos') == -1 and tweet.find('cartaaleyva') == -1 and tweet.find('cartapgn') == -1 and
tweet.find('noindultoalterrorismo') == -1 and tweet.find('acuerdodeimpunidad') == -1 and tweet.find('conclusionesforocd')
== -1 and tweet.find('noleyhabilitanteparasantos') == -1 and tweet.find('elpaisanuevohéroe') == -1 and
tweet.find('resistenciacivil') == -1 and tweet.find('santosincentivaterrorismo') == -1 and tweet.find('ptesantosnovamosbien')
== -1 and tweet.find('colombiasabedecirno') == -1 and tweet.find('decimosnoalosacuerdos') == -1 and
tweet.find('mirazónparavotarno') == -1 and tweet.find('renegociemoslosacuerdos') == -1 and tweet.find('colombiavotano')
== -1 and tweet.find('uribeargumentaporelno') == -1 and tweet.find('conargumentosdigono') == -1 and
tweet.find('hagahistoriavoteno') == -1 and tweet.find('sigueelterrorismo') == -1 and tweet.find('quierolapazvotono') == -1
and tweet.find('colombiadecideno') == -1 and tweet.find('colombiadiceno') == -1 and tweet.find('votonoycorrijoacuerdos')
== -1 and tweet.find('voteno') == -1 and tweet.find('eshoradevotarno') == -1 and tweet.find('colombiavota') == -1 and
tweet.find('elpartidodelahistoria') == -1 and tweet.find('enlineaconelpresidente') == -1 and tweet.find('cesealfuegodefinitivo')
== -1 and tweet.find('forodesminado') == -1 and tweet.find('findelconflicto') == -1 and tweet.find('fororeconciliación') == -1
and tweet.find('justiciaconlaup') == -1 and tweet.find('porlossobrevivientes') == -1):
        continue
    if (tweet.find('farc') == -1 and tweet.find('eln') is not -1):
        continue
    else:
        if (tweet.find('colombia') is not -1):
            colombiacount = colombiacount + 1
        text = df1.text[i]
        if type(text) == 'NoneType':
            continue
        else:
            text = re.sub(r"http\S+", "", text)
            text = re.sub(r"pic.twitter\S+", "", text)
            try:
                text = remove_hashtags(text)
            except:
                pass
            try:
                text = remove_mentions(text)
            except:
                pass
            if len(text) < 10 is True:
```

```
            continue
        else:
            timedate = df1.date[i]
            date = timedate
            space = date.find(' ',0)
            date = date[0:space]
            username = df1.username[i]
            permalink = df1.permalink[i]
            df2 = df2.append({'timedate' : timedate, 'date' : date , 'username' : username , 'text' : text , 'permalink' : permalink} , ignore_index=True)
            count = count + 1
            print(text)
df2.to_csv("tweets_pp.csv")
fout.write('Total number of tweets likely related to the peace process: ')
fout.write(str(count))
fout.write('\n')
fout.write('Total number of tweets containing the word FARC: ')
fout.write(str(farccount))
fout.write('\n')
fout.write('Total number of tweets containing the word Colombia: ')
fout.write(str(colombiacount))
fout.write('\n')
print('Done selecting tweets.')
```

2-translate.py

```
from google.cloud import language
from google.cloud.language import enums
from google.cloud.language import types
from google.cloud import translate_v2 as translate
import pandas as pd
from pandas import DataFrame
import re
client = language.LanguageServiceClient()
translate_client = translate.Client()
old = pd.read_csv("tweets_pp.csv")
new = old[['date', 'username', 'text', 'permalink']].copy()
total_rows = new['text'].count()
count = 0
new.insert(3, 'translation', '-')
print('Translating tweets to English...')
for i in range(0, total_rows):
    try:
```

```
        text = new.text[i]
        translation = translate_client.translate(text, target_language='en')
        transl = translation['translatedText']
        transl = re.sub('&quot;', "\"", transl)
        transl = re.sub('&#39;', "\'", transl)
        new.translation[i] = transl
        count = count + 1
    except:
        continue
new.to_csv("tweets_pp_translation.csv")
print(count)
print('Done translating tweets.')
```

3-complete_translation.py

```
from google.cloud import language
from google.cloud.language import enums
from google.cloud.language import types
from google.cloud import translate_v2 as translate
import pandas as pd
from pandas import DataFrame
import re
import os
client = language.LanguageServiceClient()
translate_client = translate.Client()
old = pd.read_csv("tweets_pp_translation.csv")
new = old.copy()
total_rows = new['text'].count()
count = 0
failed = 0
print('Idenfifying tweets that had not yet been translated and translating them...')
for i in range(0, total_rows):
    if new.translation[i] == '-':
        try:
            text = new.text[i]
            translation = translate_client.translate(text, target_language='en')
            transl = translation['translatedText']
            transl = re.sub('&quot;', "\"", transl)
            transl = re.sub('&#39;', "\'", transl)
            new.translation[i] = transl
            count = count + 1
        except:
            failed = failed + 1
```

```
print(count, ' additional translations done')
print('Failed to translate ', failed, ' tweets')
os.remove('tweets_pp_translation.csv')
new.to_csv("tweets_pp_translation.csv")
print('Your revised file is in your folder.')
```

4a-rename.py

```
import os
os.rename('output_got.csv', 'output_got-santos.csv')
os.rename('tweets_pp_translation.csv', 'tweets_pp_translation-santos.csv')
os.rename('stats.txt', 'stats-santos.txt')
os.rename('tweets_pp.csv', 'tweets_pp-santos.csv')
```

4b-rename.py

```
import os
os.rename('output_got.csv', 'output_got-uribe.csv')
os.rename('tweets_pp_translation.csv', 'tweets_pp_translation-uribe.csv')
os.rename('stats.txt', 'stats-uribe.txt')
os.rename('tweets_pp.csv', 'tweets_pp-uribe.csv')
```

5-create_corpora_entire_period.py

```
import pandas as pd
import os
print("Compiling English corpus out of peace process related tweets...")
for name in ['santos', 'uribe']:
    filename = 'corpus_all-' + name + '.txt'
    print(filename)
    fout = open(filename, 'w')
    count = 0
    file = 'tweets_pp_translation-' + name + '.csv'
    df = pd.read_csv(file)
    total_rows = df.translation.count()
    for i in range(0, total_rows):
        tweet_en = df.translation[i]
        fout.write(str(tweet_en))
        textend = tweet_en[-1:]
        if any(c.isalpha() for c in textend):
            fout.write(". ")
        else:
            fout.write(" ")
        count = count + 1
    print("English version for ", name, " compiled out of ", str(count), "tweets")
print('Done compiling English corpora for the entire period.')
```

6-create_corpora_fighting_vs_ceasefire.py

```
import pandas as pd
import os
from datetime import date
statsfile = open('stats-noCF-vs-CF-periods.txt', 'w')
print('Compiling corpus for Santos during fighting periods')
fout = open('corpus_noCF-santos.txt', 'w')
count = 0
noCFdaycount = 0
for period in ['1', '3', '5', '7', '9', '11']:
    if period == '1':
        start = '2012-08-26'
        end = '2012-11-18'
    if period == '2':
        start = '2012-11-18'
        end = '2013-01-18'
    if period == '3':
        start = '2013-01-18'
        end = '2013-12-07'
    if period == '4':
        start = '2013-12-07'
        end = '2014-01-07'
    if period == '5':
        start = '2014-01-07'
        end = '2014-05-19'
    if period == '6':
        start = '2014-05-19'
        end = '2014-05-28'
    if period == '7':
        start = '2014-05-28'
        end = '2014-06-08'
    if period == '8':
        start = '2014-06-08'
        end = '2014-06-30'
    if period == '9':
        start = '2014-06-30'
        end = '2014-12-19'
    if period == '10':
        start = '2014-12-19'
        end = '2015-05-20'
    if period == '11':
```

```
        start = '2015-05-20'
        end = '2015-07-19'
    if period == '12':
        start = '2015-07-19'
        end = '2016-10-02'
    startyear = start[0:4]
    startmonth = start[5:7]
    startday = start[8:]
    if startmonth[0:1] == "0":
        startmonth = startmonth[1:2]
    if startday[0:1] == "0":
        startday = startday[1:2]
    endyear = end[0:4]
    endmonth = end[5:7]
    endday = end[8:]
    if endmonth[0:1] == "0":
        endmonth = endmonth[1:2]
    if endday[0:1] == "0":
        endday = endday[1:2]
    delta = date(int(endyear), int(endmonth), int(endday)) - date(int(startyear), int(startmonth), int(startday))
    noCFdaycount = noCFdaycount + delta.days
    data = pd.read_csv('tweets_pp_translation-santos.csv')
    total_tweets = data.text.count()
    mask = (data['date'] > start) & (data['date'] <= end)
    df = data.loc[mask]
    df.to_csv("df.csv")
    df = pd.read_csv("df.csv")
    total_rows = df.text.count()
    for i in range(0, total_rows):
        tweet_en = df.translation[i]
        fout.write(str(tweet_en))
        textend = tweet_en[-1:]
        if any(c.isalpha() for c in textend):
            fout.write(". ")
        else:
            fout.write(" ")
        count = count + 1
    os.remove("df.csv")
print(str(count), ' tweets included in corpus Santos no CF')
statsfile.write("\n")
statsfile.write("Santos total pp related tweets: ")
```

```
statsfile.write(str(total_tweets))
statsfile.write("\n")
statsfile.write(str(count))
statsfile.write(" during fighting")
statsfile.write("\n")
print('Compiling corpus for Santos during ceasefire periods')
fout = open('corpus_CF-santos.txt', 'w')
count = 0
CFdaycount = 0
for period in ['2', '4', '6', '8', '10', '12']:
[add periods]
    startyear = start[0:4]
    startmonth = start[5:7]
    startday = start[8:]
    if startmonth[0:1] == "0":
        startmonth = startmonth[1:2]
    if startday[0:1] == "0":
        startday = startday[1:2]
    endyear = end[0:4]
    endmonth = end[5:7]
    endday = end[8:]
    if endmonth[0:1] == "0":
        endmonth = endmonth[1:2]
    if endday[0:1] == "0":
        endday = endday[1:2]
    delta = date(int(endyear), int(endmonth), int(endday)) - date(int(startyear), int(startmonth), int(startday))
    CFdaycount = CFdaycount + delta.days
    data = pd.read_csv('tweets_pp_translation-santos.csv')
    mask = (data['date'] > start) & (data['date'] <= end)
    df = data.loc[mask]
    df.to_csv("df.csv")
    df = pd.read_csv("df.csv")
    total_rows = df.text.count()
    for i in range(0, total_rows):
        tweet_en = df.translation[i]
        fout.write(str(tweet_en))
        textend = tweet_en[-1:]
```

[116] To save space, [add periods] is used here as a placeholder to indicate that the start and end dates for the periods should be added (as at the beginning of the script).

```
        if any(c.isalpha() for c in textend):
            fout.write(". ")
        else:
            fout.write(" ")
        count = count + 1
    os.remove("df.csv")
print(str(count), ' tweets included in corpus Santos CF')
statsfile.write(str(count))
statsfile.write(" during CF")
statsfile.write("\n")
statsfile.write("\n")
statsfile.write("\n")
print('Compiling corpus for Uribe during fighting periods')
fout = open('corpus_noCF-uribe.txt', 'w')
count = 0
for period in ['1', '3', '5', '7', '9', '11']:
    [add periods]
    data = pd.read_csv('tweets_pp_translation-uribe.csv')
    total_tweets = data.text.count()
    mask = (data['date'] > start) & (data['date'] <= end)
    df = data.loc[mask]
    df.to_csv("df.csv")
    df = pd.read_csv("df.csv")
    total_rows = df.text.count()
    for i in range(0, total_rows):
        tweet_en = df.translation[i]
        fout.write(str(tweet_en))
        textend = tweet_en[-1:]
        if any(c.isalpha() for c in textend):
            fout.write(". ")
        else:
            fout.write(" ")
        count = count + 1
    os.remove("df.csv")
print(str(count), ' tweets included in corpus Uribe no CF')
statsfile.write("Uribe total pp related tweets: ")
statsfile.write(str(total_tweets))
statsfile.write("\n")
statsfile.write(str(count))
statsfile.write(" during fighting")
statsfile.write("\n")
```

```
print('Compiling corpus for Uribe during ceasefire periods')
fout = open('corpus_CF-uribe.txt', 'w')
count = 0
for period in ['2', '4', '6', '8', '10', '12']:
    [add periods]
    data = pd.read_csv('tweets_pp_translation-uribe.csv')
    mask = (data['date'] > start) & (data['date'] <= end)
    df = data.loc[mask]
    df.to_csv("df.csv")
    df = pd.read_csv("df.csv")
    total_rows = df.text.count()
    for i in range(0, total_rows):
        tweet_en = df.translation[i]
        fout.write(str(tweet_en))
        textend = tweet_en[-1:]
        if any(c.isalpha() for c in textend):
            fout.write(". ")
        else:
            fout.write(" ")
        count = count + 1
    os.remove("df.csv")
print(str(count), ' tweets included in corpus Uribe CF')
statsfile.write(str(count))
statsfile.write(" during ceasefire")
statsfile.write("\n")
statsfile.write("\n")
statsfile.write("\n")
statsfile.write("\n")
totaldays = noCFdaycount + CFdaycount
statsfile.write(str(totaldays))
statsfile.write(" days in entire period")
statsfile.write("\n")
statsfile.write(str(noCFdaycount))
statsfile.write(" days of fighting")
statsfile.write("\n")
statsfile.write(str(CFdaycount))
statsfile.write(" days of ceasefire")
statsfile.write("\n")
print('Done compiling.')
```

7-identifying_past_tense.py

```
import spacy
```

```
import pandas as pd
from pandas import DataFrame
import os
nlp = spacy.load("en_core_web_sm")
def pastcount(text):
    sentencecount = 0
    pastcount = 0
    df_temp = DataFrame(columns = ['sentencecount', 'pastcount'])
    for sent in doc.sents:
        if len(sent) < 10:
            continue
        else:
            sentencecount = sentencecount + 1
            pastform = 0
            dependencies = list(sent.root.children)
            for word in sent:
                if sent.root.tag_ == 'VBD':
                    pastform = pastform + 1
                else:
                    if word in dependencies:
                        if word.dep_ == 'aux' and word.tag_ == 'VBD':
                            pastform = pastform + 1
            if pastform is not 0:
                pastcount = pastcount + 1
                print(sent)
    df_temp = df_temp.append({'sentencecount' : sentencecount, 'pastcount' : pastcount} , ignore_index=True)
    df_temp.to_csv("df_temp.csv")
for name in ['santos', 'uribe']:
    for period in ['_all', '_noCF', '_CF']:
        filename = "corpus" + str(period) + "-" + str(name) + ".txt"
        corpus = open(filename).read()
        doc = nlp(corpus)
        pastcount(doc)
        outputname = "corpus" + str(period) + "-" + str(name) + "-past-stats.csv"
        os.rename('df_temp.csv', outputname)
df = pd.DataFrame(columns=['name', 'period', 'sentencecount', 'pastcount', 'percentage'])
for name in ['santos', 'uribe']:
    for period in ['_all', '_noCF', '_CF']:
        filename = "corpus" + str(period) + "-" + str(name) + "-past-stats.csv"
        data = pd.read_csv(filename)
        sentencecount = data.sentencecount[0]
```

```
        pastcount = data.pastcount[0]
        percentage = int(pastcount)/int(sentencecount)
        percentage = '{:.0%}'.format(percentage)
        df = df.append({'name' : name, 'period' : period, 'sentencecount' : sentencecount, 'pastcount' : pastcount, 'percentage' : percentage} , ignore_index=True)
        os.remove(filename)
df.to_csv("stats-past_tense.csv")
print('Done.')
```

8-identifying_future_tense.py

```
import spacy
import pandas as pd
import os
nlp = spacy.load("en_core_web_sm")
from pandas import DataFrame
def futurecount(text):
    sentencecount = 0
    futurecount = 0
    df_temp = DataFrame(columns = ['sentencecount', 'futurecount'])
    for sent in doc.sents:
        if len(sent) < 10:
            continue
        else:
            sentencecount = sentencecount + 1
            auxcount = 0
            for word in sent:
                if word.dep_ == 'aux':
                    if word.text in ['will', 'shall']:
                        auxcount = auxcount + 1
            if auxcount is not 0:
                futurecount = futurecount + 1
                print(sent)
    df_temp = df_temp.append({'sentencecount' : sentencecount, 'futurecount' : futurecount} , ignore_index=True)
    df_temp.to_csv("df_temp.csv")
for name in ['santos', 'uribe']:
    for period in ['_all', '_noCF', '_CF']:
        filename = "corpus" + str(period) + "-" + str(name) + ".txt"
        corpus = open(filename).read()
        doc = nlp(corpus)
        futurecount(doc)
        outputname = "corpus" + str(period) + "-" + str(name) + "-future-stats.csv"
        os.rename('df_temp.csv', outputname)
```

```
df = pd.DataFrame(columns=['name', 'period', 'sentencecount', 'futurecount', 'percentage'])
for name in ['santos', 'uribe']:
    for period in ['_all', '_noCF', '_CF']:
        filename = "corpus" + str(period) + "-" + str(name) + "-future-stats.csv"
        data = pd.read_csv(filename)
        sentencecount = data.sentencecount[0]
        futurecount = data.futurecount[0]
        percentage = int(futurecount)/int(sentencecount)
        percentage = '{:.0%}'.format(percentage)
        df = df.append({'name' : name, 'period' : period, 'sentencecount' : sentencecount, 'futurecount' : futurecount, 'percentage'
: percentage} , ignore_index=True)
        os.remove(filename)
df.to_csv("stats-future_tense.csv")
print('Done.')
```

9-stops_and_lemma.py

```
import pandas as pd
import re
import string
import nltk
from nltk.corpus import stopwords
import os
import spacy
for period_name in ['all-santos', 'noCF-santos', 'CF-santos', 'all-uribe', 'noCF-uribe', 'CF-uribe']:
    file = "corpus_" + str(period_name) + ".txt"
    text = open(file, 'r').read()
    text = text.lower()
    text = re.sub('&#39;', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\w*\d\w*', '', text)
    text = re.sub('[‘’“”…]', '', text)
    text = re.sub('\n', '', text)
    text = re.sub('&quot;', '', text)
    text = re.sub('quot', '', text)
    text = re.sub('#', '', text)
    text = re.sub('guerra', 'war', text)
    text = re.sub('paz ', 'peace ', text)
    text = ' '.join([word for word in text.split() if word not in (stopwords.words('english'))])
    nlp = spacy.load('en_core_web_sm')
    count = 0
    output = "corpus-lemma-" + str(period_name) + ".txt"
    fout = open(output, 'w+')
```

```
    doc = nlp(text)
    count = 0
    out_sent = [w.lemma_ if w.lemma_ !='-PRON-' else w.text for w in doc]
    out_sent = ' '.join(out_sent)
    out_sent = re.sub('santo ', "santos ", out_sent) #corrects for falsly lemmatized version of Santos
    print(out_sent)
    fout.write(out_sent + '\n')
    print(count)
    fout.close()
print('Done. Your lemmatized corpora are in your folder')
```

10-word_frequencies-wordcloud.py

```
from collections import Counter
import re
from wordcloud import WordCloud
wc = WordCloud(normalize_plurals=False, background_color="white", color_func=lambda *args, **kwargs: "black", max_font_size=120, random_state=42)
import matplotlib.pyplot as plt
fout = open("most_frequent_words.txt", "w")
for period_name in ['all-santos', 'noCF-santos', 'CF-santos', 'all-uribe', 'noCF-uribe', 'CF-uribe']:
    short = "corpus-lemma-" + str(period_name)
    filename = short + ".txt"
    words = re.findall(r'\w+', open(filename).read())
    fout.write(period_name)
    fout.write("\n")
    fout.write("\n")
    fout.write(str(Counter(words).most_common(300)))
    fout.write("\n")
    fout.write("\n")
    fout.write("\n")
    text = open(filename).read()
    wc.generate(text)
    plt.imshow(wc, interpolation="bilinear")
    plt.axis("off")
    plt.savefig(short)
print('Most frequent words are in a textfile in your folder, wordclouds saved as graphs.')
```

11-create_order.py

```
import os
import shutil
os.makedirs('corpora-santos')
os.makedirs('corpora-uribe')
os.makedirs('tweets')
```

```
os.makedirs('graphs')
os.makedirs('stats')
path = os.getcwd()
shutil.move(path + "/corpus_all-santos.txt", path + '/corpora-santos')
shutil.move(path + "/corpus_noCF-santos.txt", path + '/corpora-santos')
shutil.move(path + "/corpus_CF-santos.txt", path + '/corpora-santos')
shutil.move(path + "/corpus-lemma-all-santos.txt", path + '/corpora-santos')
shutil.move(path + "/corpus-lemma-noCF-santos.txt", path + '/corpora-santos')
shutil.move(path + "/corpus-lemma-CF-santos.txt", path + '/corpora-santos')
shutil.move(path + "/corpus_all-uribe.txt", path + '/corpora-uribe')
shutil.move(path + "/corpus_noCF-uribe.txt", path + '/corpora-uribe')
shutil.move(path + "/corpus_CF-uribe.txt", path + '/corpora-uribe')
shutil.move(path + "/corpus-lemma-all-uribe.txt", path + '/corpora-uribe')
shutil.move(path + "/corpus-lemma-noCF-uribe.txt", path + '/corpora-uribe')
shutil.move(path + "/corpus-lemma-CF-uribe.txt", path + '/corpora-uribe')
shutil.move(path + "/corpus-lemma-all-santos.png", path + '/graphs')
shutil.move(path + "/corpus-lemma-noCF-santos.png", path + '/graphs')
shutil.move(path + "/corpus-lemma-CF-santos.png", path + '/graphs')
shutil.move(path + "/corpus-lemma-all-uribe.png", path + '/graphs')
shutil.move(path + "/corpus-lemma-noCF-uribe.png", path + '/graphs')
shutil.move(path + "/corpus-lemma-CF-uribe.png", path + '/graphs')
shutil.move(path + "/stats-future_tense.csv", path + '/stats')
shutil.move(path + "/stats-past_tense.csv", path + '/stats')
shutil.move(path + "/stats-noCF-vs-CF-periods.txt", path + '/stats')
shutil.move(path + "/stats-santos.txt", path + '/stats')
shutil.move(path + "/stats-uribe.txt", path + '/stats')
shutil.move(path + "/most_frequent_words.txt", path + '/stats')
shutil.move(path + "/tweets_pp_translation-santos.csv", path + '/tweets')
shutil.move(path + "/tweets_pp_translation-uribe.csv", path + '/tweets')
shutil.move(path + "/tweets_pp-santos.csv", path + '/tweets')
shutil.move(path + "/tweets_pp-uribe.csv", path + '/tweets')
shutil.move(path + "/output_got-santos.csv", path + '/tweets')
shutil.move(path + "/output_got-uribe.csv", path + '/tweets')
print('Done. All your files are in folders.')
```