Installation

```
pip install google-pygram
```

```
Collecting google-pygram
  Downloading google_pygram-0.0.1-py3-none-any.whl (4.6 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from google-pygram) (2.31.0)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from google-pygram) (1.5.3)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas->google-pygram) (2.8.
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->google-pygram) (2023.3.post1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from pandas->google-pygram) (1.23.5)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->google-pygram) (
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->google-pygram) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->google-pygram) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->google-pygram) (2023.7
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.1->pandas->google-pyg
Installing collected packages: google-pygram
Successfully installed google-pygram-0.0.1
```

```
import pandas as pd
import numpy as np
```

tdt4117-assist@idi.ntnu.no

Part 1

Visualizing the frequency of terms in google_pygram.

Assign the duration of search

```
search_strat_year = 1800
search_end_year = 2019
```

```
windows_phrases = ["Windows *"]
```

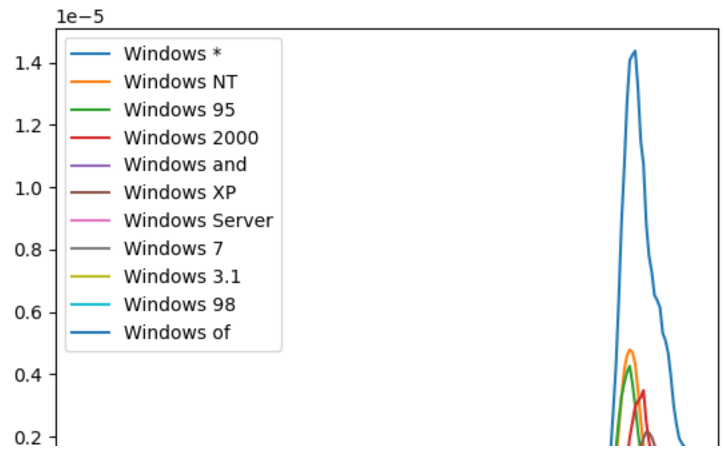
```
from google_pygram import GooglePyGram as gpg

# get the pygram
pygram = gpg(
    corpus='English',
    corpus_year=2019,
    start_year=search_strat_year,
    end_year=search_end_year,
    smoothing=3,
    case_sensitive=False,
    phrases=windows_phrases
)
```

```
windows_ngram = pygram.to_df()
windows_ngram
```

	year	Windows *	Windows NT	Windows 95	Windows 2000	Windows and	Windows XP	Windows Server	Windows 7	Windows
1800	1800.0	1.256992e-07	0.000000e+00	0.000000e+00	0.000000e+00	6.618923e-08	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e

```
# just plot
windows_ngram.plot(x='year')
```

```
<Axes: xlabel='year'>
```

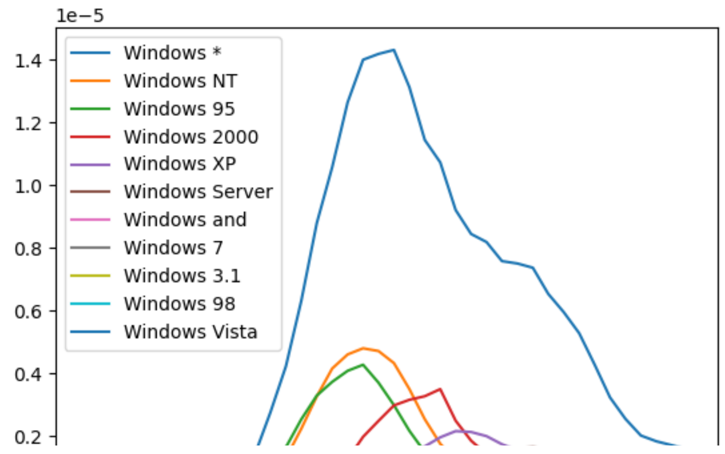


Now we get the time periods between 1980 to 2019

```
search_strat_year = 1980
search_end_year = 2019

# get the pygram
pygram = gpg(
    corpus='English',
    corpus_year=2019,
    start_year=search_strat_year,
    end_year=search_end_year,
    smoothing=3,
    case_sensitive=False,
    phrases=windows_phrases
)

# convert to dataframe
windows_ngram = pygram.to_df()
```

```
windows_ngram.plot(x='year')
```

```
<Axes: xlabel='year'>
```

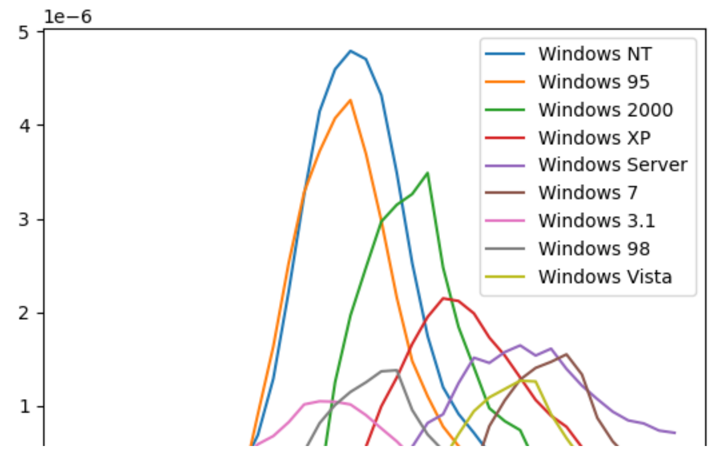


Pre-process the dataframe

```
# we drop the iPhone * and iPhone and to pre process the dataframe
windows_ngram = windows_ngram.drop(
    columns = ['Windows *', 'Windows and'])
```

```
windows_ngram.plot(x="year")
```

```
<Axes: xlabel='year'>
```



Part 2

Visualizing the results to see the relevance

```
search_strat_year = 1990
search_end_year = 2019
```

Assign list the phrases to search

```
windows_phrases = ["Windows *"]
```

Now we get the frequency of the terms from GooglePygram. Then convert it to a dataframe.

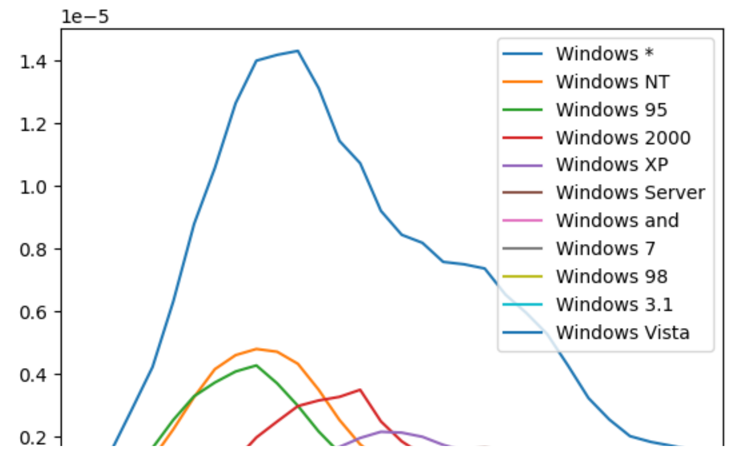
```
from google_pygram import GooglePyGram as gpg

# get the pygram
pygram = gpg(
    corpus='English',
    corpus_year=2019,
    start_year=search_strat_year,
    end_year=search_end_year,
    smoothing=3,
    case_sensitive=False,
    phrases=windows_phrases
)
```

convert to the dataframe and pre process

```
windows_ngram = pygram.to_df()
```

```
windows_ngram.plot(x="year")
```

```
<Axes: xlabel='year'>
```



Part 3

Computational approach:

We calculate the dissimilarities between Windows Vista and Windows NT

```
dissimilarity = np.log(windows_ngram['Windows Vista'] / windows_ngram['Windows NT']) * windows_ngram['Windows Vista']
```

```
pd.DataFrame(dissimilarity)
```

Let us visualize the dataframe.

Part 4

Now use the computation method to calculate the time period of dissimilarities between two terms

French presidents:

Tip: use start year of 2000 and end year of 2019

```
# Have your code here
search_strat_year = 2000
search_end_year = 2019

french_president_phrases = ["French President *"]

# get the pygram
pygram = gpg(
    corpus='English',
    corpus_year=2019,
    start_year=search_strat_year,
    end_year=search_end_year,
    smoothing=3,
    case_sensitive=False,
    phrases=french_president_phrases
)

french_president_ngram = pygram.to_df()
french_president_ngram

# we drop the stopwords "*", "and", "'s" to pre process the dataframe
french_president_ngram = french_president_ngram.drop(
    columns = ['French President *', 'French President and', """French President 's"""])

french_president_ngram.plot(x="year")
french_president_ngram
```

	year	French President Jacques	French President Nicolas	French President Charles	French President François	French Preside
2000	2000.0	3.578562e-08	8.459316e-11	9.695043e-09	2.121189e-09	
2001	2001.0	3.816958e-08	8.897159e-11	9.473998e-09	2.484942e-09	
2002	2002.0	3.996282e-08	1.305128e-10	9.442036e-09	2.427794e-09	
2003	2003.0	3.935524e-08	1.943983e-10	9.485871e-09	2.369822e-09	
2004	2004.0	3.915010e-08	1.904471e-09	9.740633e-09	2.404678e-09	
2005	2005.0	3.777354e-08	5.421427e-09	9.364873e-09	2.616921e-09	
2006	2006.0	3.588870e-08	8.745559e-09	9.644675e-09	3.031636e-09	
2007	2007.0	3.062521e-08	1.260542e-08	9.717940e-09	3.405515e-09	
2008	2008.0	2.540806e-08	1.570666e-08	9.448509e-09	3.435264e-09	
2009	2009.0	1.993662e-08	1.877814e-08	9.255065e-09	3.788897e-09	
2010	2010.0	1.660378e-08	2.119698e-08	9.233823e-09	4.482576e-09	
2011	2011.0	1.325700e-08	2.146811e-08	9.116216e-09	5.475810e-09	
2012	2012.0	1.185176e-08	2.003860e-08	8.974023e-09	6.605609e-09	
2013	2013.0	1.147123e-08	1.928281e-08	8.971555e-09	8.183575e-09	
2014	2014.0	1.081618e-08	1.693493e-08	8.784764e-09	9.155565e-09	
2015	2015.0	1.021752e-08	1.494806e-08	8.864337e-09	1.020303e-08	
2016	2016.0	9.437605e-09	1.283587e-08	8.870792e-09	1.117685e-08	
2017	2017.0	8.947215e-09	1.205692e-08	8.749226e-09	1.189337e-08	
2018	2018.0	8.936336e-09	1.168658e-08	8.607759e-09	1.230924e-08	
2019	2019.0	8.647338e-09	1.094508e-08	8.460889e-09	1.246164e-08	

```
from scipy.special import rel_entr

# For french president instances I choose "Jacques" and "Nicolas"
Jaques = [3.578562e-08, 3.816958e-08, 3.996282e-08, 3.935524e-08, 3.915010e-08, 3.777354e-08, 3.588870e-08, 3.062521e-08, 2.540806e-08, 1
Nicolas = [8.459316E-11, 8.897159E-11, 0.0000000001305128, 0.0000000001943983, 0.000000001904471, 0.000000005421427, 0.000000008745559, 0


#calculate (P || Q), method from https://www.statology.org/kl-divergence-python/
sum(rel_entr(Jaques, Nicolas))


# We can see that Jacques is higher than Nicolas from year 2000 until 2009, then from 2010 until 2019 Jacques is higher than Nicolas
```

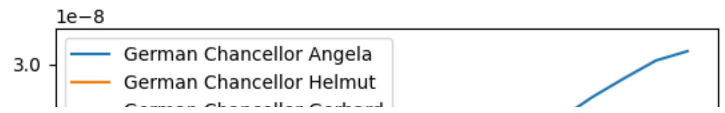
```
1.1271129399021164e-06
```

German Chancellors:

Tip: use start year of 2000 and end year of 2019

```
# Have your code here

search_strat_year = 2000
search_end_year = 2019

german_chancellor_phrases = ["German Chancellor *"]

# get the pygram
pygram = gpg(
    corpus='English',
    corpus_year=2019,
    start_year=search_strat_year,
    end_year=search_end_year,
    smoothing=3,
    case_sensitive=False,
    phrases=german_chancellor_phrases
)

german_chancellor_ngram = pygram.to_df()

# we drop the stopwords "*", "and", "'s", "in" to pre process the dataframe
german_chancellor_ngram = german_chancellor_ngram.drop(
    columns = ['German Chancellor *', 'German Chancellor in', """German Chancellor 's""", 'German Chancellor and'])

german_chancellor_ngram.plot(x="year")
german_chancellor_ngram
```

	year	German Chancellor Angela	German Chancellor Helmut	German Chancellor Gerhard	German Chancellor Willy	German Chancell
2000	2000.0	1.632169e-10	1.687226e-08	2.514413e-08	3.905666e-09	4.
2001	2001.0	1.660686e-10	1.682015e-08	2.633405e-08	4.175188e-09	4.
2002	2002.0	5.956123e-10	1.615258e-08	2.654858e-08	4.174551e-09	3.
2003	2003.0	2.388087e-09	1.562978e-08	2.499971e-08	4.104317e-09	3.
2004	2004.0	5.324471e-09	1.497863e-08	2.365287e-08	4.200046e-09	4.
2005	2005.0	7.908253e-09	1.404574e-08	2.159878e-08	4.021131e-09	4.
2006	2006.0	1.029516e-08	1.371167e-08	1.867593e-08	4.186653e-09	4.
2007	2007.0	1.290464e-08	1.370637e-08	1.556959e-08	4.246540e-09	4.
2008	2008.0	1.520737e-08	1.271953e-08	1.183134e-08	4.025180e-09	4.
2009	2009.0	1.741253e-08	1.229526e-08	8.569985e-09	3.864881e-09	4.
2010	2010.0	1.840070e-08	1.220951e-08	7.251416e-09	3.973305e-09	4.
2011	2011.0	1.873610e-08	1.171460e-08	6.288270e-09	4.097789e-09	4.
2012	2012.0	2.006914e-08	1.146647e-08	5.600069e-09	4.310988e-09	4.
2013	2013.0	2.286174e-08	1.160122e-08	5.415194e-09	4.304847e-09	4.
2014	2014.0	2.409713e-08	1.118689e-08	5.068942e-09	4.432330e-09	4.
2015	2015.0	2.627108e-08	1.127777e-08	4.930018e-09	4.332245e-09	4.
2016	2016.0	2.774202e-08	1.190537e-08	4.664805e-09	4.610101e-09	3.
2017	2017.0	2.902234e-08	1.190747e-08	4.362848e-09	4.638139e-09	3.
2018	2018.0	3.023827e-08	1.209219e-08	4.273513e-09	4.564865e-09	3.
2019	2019.0	3.088272e-08	1.210622e-08	4.155108e-09	4.341748e-09	3.



```
from scipy.special import rel_entr

# For french president instances I choose "Jacques" and "Nicolas"
Angela = [0.0000000001632169,
0.0000000001660686,
0.0000000005956123,
0.000000002388087,
0.000000005324471,
0.000000007908253,
0.00000001029516,
0.00000001290464,
0.00000001520737,
0.00000001741253,
0.0000000184007,
0.0000000187361,
0.00000002006914,
0.00000002286174,
0.00000002409713,
0.00000002627108,
0.00000002774202,
0.00000002902234,
0.00000003023827,
3.088272E-08]


Helmut = [0.00000001687226,
0.00000001682015,
0.00000001615258,
0.00000001562978,
0.00000001497863,
0.00000001404574,
0.00000001371167,
0.00000001370637,
0.00000001271953,
0.00000001229526,
0.00000001220951,
0.0000000117146,
0.00000001146647,
0.00000001160122,
0.00000001118689,
0.00000001127777,
0.00000001190537,
0.00000001190747,
0.00000001209219,
1.210622E-08]


#calculate (P || Q), method from https://www.statology.org/kl-divergence-python/
sum(rel_entr(Angela, Helmut))


# We can see that Helmut is higher than Angela from year 2000 until 2007, then from 2008 until 2019 Angela is higher than Helmut
```

```
1.767773967435491e-07
```

War in:

Tip: use start year of 1940 and end year of 2019

```
# Have your code here

search_strat_year = 2000
search_end_year = 2019

war_in_phrases = ["War in *"]

# get the pygram
pygram = gpg(
    corpus='English',
    corpus_year=2019,
    start_year=search_strat_year,
    end_year=search_end_year,
    smoothing=3,
    case_sensitive=False,
    phrases=war_in_phrases
)

war_in_ngram = pygram.to_df()

# we drop the stopwords "*", "and", "'s" to pre process the dataframe
war_in_ngram = war_in_ngram.drop(
    columns = ['War in *', 'War in the', 'War in a'])

war_in_ngram.plot(x="year")
war_in_ngram
```

	year	War in Europe	War in Iraq	War in Vietnam	War in 1991	War in South	War in Afghanistan	War in America	War in France
2000	2000.0	6.785493e-08	2.362792e-08	5.642062e-08	5.299976e-08	3.464327e-08	2.344082e-08	5.128624e-08	2.294899e-08
2001	2001.0	7.016446e-08	3.608197e-08	5.475567e-08	5.632099e-08	3.380557e-08	2.266422e-08	4.667981e-08	2.225656e-08
2002	2002.0	7.000114e-08	4.582054e-08	5.484306e-08	5.780727e-08	3.271848e-08	2.255715e-08	4.376644e-08	2.207964e-08
2003	2003.0	7.006450e-08	5.253116e-08	5.621045e-08	5.737914e-08	3.262484e-08	2.207883e-08	4.220799e-08	2.242778e-08
2004	2004.0	7.320740e-08	6.895275e-08	5.643116e-08	5.926008e-08	3.203835e-08	2.371825e-08	3.789383e-08	2.319421e-08
2005	2005.0	7.433431e-08	8.562501e-08	5.896147e-08	6.125052e-08	3.162808e-08	2.471266e-08	3.645602e-08	2.383961e-08
2006	2006.0	7.436368e-08	1.009824e-07	5.950423e-08	5.941379e-08	3.189910e-08	2.215169e-08	3.258672e-08	2.375147e-08
2007	2007.0	7.481118e-08	1.040828e-07	6.247899e-08	5.649008e-08	3.187295e-08	2.375342e-08	3.303456e-08	2.394879e-08
2008	2008.0	7.473267e-08	1.034467e-07	6.627497e-08	5.257292e-08	3.207260e-08	2.817648e-08	3.299831e-08	2.640842e-08
2009	2009.0	7.664800e-08	9.827953e-08	6.652998e-08	4.784570e-08	3.252284e-08	3.157712e-08	3.253172e-08	2.725128e-08
2010	2010.0	7.865328e-08	9.462330e-08	6.767723e-08	4.638166e-08	3.369843e-08	3.638214e-08	3.116615e-08	2.878439e-08
2011	2011.0	8.051601e-08	8.533879e-08	6.903747e-08	4.313385e-08	3.393089e-08	3.902583e-08	2.805432e-08	2.970568e-08
2012	2012.0	8.349487e-08	7.693909e-08	6.952634e-08	4.101379e-08	3.502849e-08	4.272980e-08	2.760279e-08	3.211553e-08
2013	2013.0	8.753265e-08	7.017671e-08	7.024918e-08	4.024826e-08	3.801820e-08	4.604675e-08	2.658905e-08	3.475038e-08
2014	2014.0	8.728896e-08	6.261503e-08	6.878061e-08	3.806030e-08	3.800046e-08	4.646326e-08	2.575393e-08	3.647791e-08
2015	2015.0	8.527811e-08	5.769768e-08	6.776953e-08	3.584066e-08	3.748512e-08	4.464975e-08	2.473531e-08	3.639902e-08
2016	2016.0	8.336768e-08	5.387536e-08	6.879231e-08	3.485922e-08	3.745566e-08	4.358557e-08	2.418481e-08	3.913100e-08
2017	2017.0	8.318202e-08	5.165436e-08	6.818343e-08	3.324208e-08	3.698291e-08	4.204249e-08	2.433256e-08	3.977809e-08
2018	2018.0	8.059333e-08	5.153427e-08	6.790500e-08	3.286469e-08	3.691823e-08	4.289224e-08	2.411657e-08	4.144884e-08

```
from scipy.special import rel_entr

# For french president instances I choose "Jacques" and "Nicolas"
Europe = [0.00000006785493,
0.00000007016446,
0.00000007000114,
0.0000000700645,
0.0000000732074,
0.00000007433431,
0.00000007436368,
0.00000007481118,
0.00000007473267,
0.000000076648,
0.00000007865328,
0.00000008051601,
0.00000008349487,
0.00000008753265,
0.00000008728896,
0.00000008527811,
0.00000008336768,
0.00000008318202,
0.00000008059333,
7.591265E-08]


Iraq = [0.00000002362792,
0.00000003608197,
0.00000004582054,
0.00000005253116,
0.00000006895275,
0.00000008562501,
0.0000001009824,
0.0000001040828,
0.0000001034467,
0.00000009827953,
0.0000000946233,
0.00000008533879,
0.00000007693909,
0.00000007017671,
0.00000006261503,
0.00000005769768,
0.00000005387536,
0.00000005165436,
0.00000005153427,
4.898031E-08]


#calculate (P || Q), method from https://www.statology.org/kl-divergence-python/
sum(rel_entr(Europe, Iraq))


# We can see that Europe is higher than Iraq from year 2000 until 2004, then from 2005 until 2011 Iraq is higher than Europe, then from 2
```

```
2.857485490563804e-07
```

```
import random; random.seed(123)
```

```
import codecs
f = codecs.open("pg3300.txt", "r", "utf-8")
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-43-a89a55861d81> in <cell line: 2>()
      1 import codecs
----> 2 f = codecs.open("pg3300.txt", "r", "utf-8")

/usr/lib/python3.10/codecs.py in open(filename, mode, encoding, errors, buffering)
    904         # Force opening of the file in binary mode
    905         mode = mode + 'b'
--> 906     file = builtins.open(filename, mode, buffering)
    907     if encoding is None:
    908         return file

FileNotFoundError: [Errno 2] No such file or directory: 'pg3300.txt'
```

SEARCH STACK OVERFLOW

Double-click (or enter) to edit