



FALL SEMESTER 2025-26

CSI3007- Advance Python Programming
SLOT: L7+L8+L27+L28

LAB DIGITAL ASSIGNMENT – 10

Topic: Explore “Forgetting” trends in interesting schools of thought that influenced lives over the past two centuries, using Google Books Ngram Viewer. Extract the plots, digitize the data, analyze with LLM support, and present insights.

Faculty: Dr. Sharmila Banu K

NAME: Praveen Kumar G

REG NO.: 22MID0300

Date: 29-08-2025

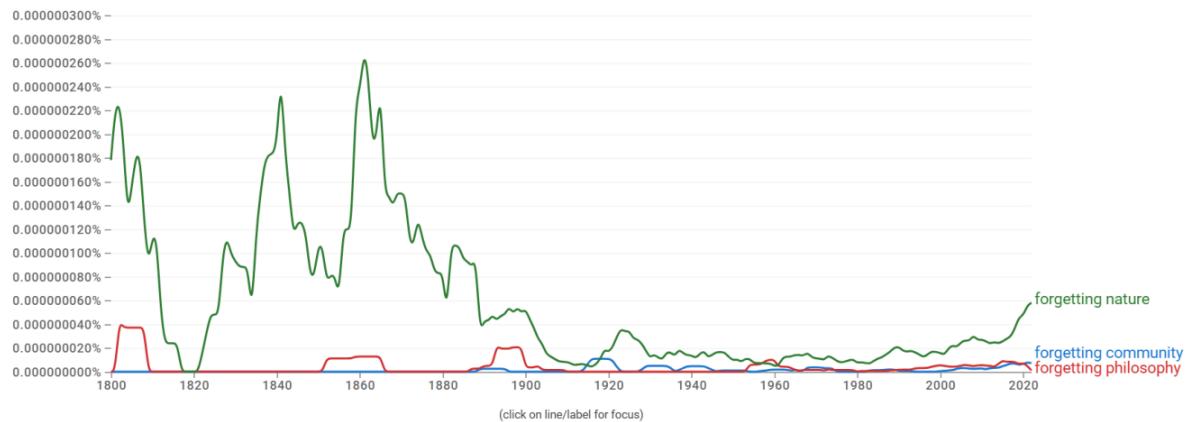
Objective

The objective of this lab is to analyze the trend of “Forgetting” in selected schools of thought that influenced lives in the past two centuries. Using Google Books Ngram Viewer, graph data was extracted, digitized, and analyzed with the help of data processing tools and Large Language Models (LLMs).

Methodology

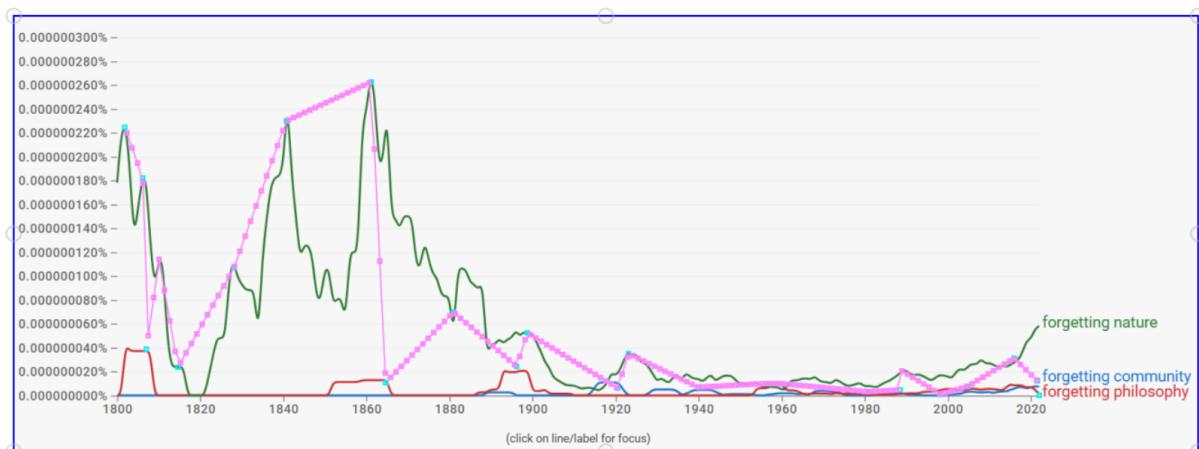
1. Data Collection

- Keywords chosen: *Forgetting Nature*, *Forgetting Community*, *Forgetting Philosophy*.
- Google Books Ngram Viewer used to generate graphs (1800–2020).



2. Data Extraction

- Graph images were digitized using **WebPlotDigitizer**.
- X-axis calibrated (Years: 1800–2020).
- Y-axis calibrated (Relative frequency values: 0 - 0.00000300).
- CSV data exported for each curve.



Input Values:

Generate curve ► Clear fix-points ✖ Redraw fix-points ⚡ Redraw sampled curve 🌸 Redraw post processed 🌿 Stats

Axis Settings (defined by blue rectangle)

y-high: 0.00000301	X-axis scaling (for dates use format: YYYYMMDD)	<input checked="" type="radio"/> X-axis stepsize	<input type="radio"/> Number of intervals (not dates)
y-low: 0	x-low: 1800 x-high: 2020	value: 1	Snap fix-points to x-values
<input type="checkbox"/> Log y-axis (beta)	<input type="checkbox"/> Log x-axis (beta)		

Curve Sampling (using fix-points) Post-processing

<input checked="" type="radio"/> Point interpolation	<input type="radio"/> Machine estimation	<input checked="" type="radio"/> None	
<input checked="" type="radio"/> Linear	<input checked="" type="radio"/> Standard	<input type="radio"/> Average	Linear Regression, order: 1
<input type="radio"/> Spline	<input type="radio"/> Smooth curves	<input type="radio"/> Running Average	Moving Average, length: 5
	<input type="radio"/> Steep curves	<input type="radio"/> Spline approximation, order: 3	

Data Output

Precision, output decimals: 3	CSV delimiter: ,	<input checked="" type="checkbox"/> Show data in new window (Blob)														
Inserted Fix-Points																
<table border="1"> <tr><td>1820.541</td><td>0</td></tr> <tr><td>1823.979</td><td>0</td></tr> <tr><td>1824.649</td><td>0</td></tr> <tr><td>1826.997</td><td>0</td></tr> <tr><td>1830.518</td><td>0</td></tr> <tr><td>1840.915</td><td>0</td></tr> <tr><td>1850.64</td><td>0</td></tr> </table>			1820.541	0	1823.979	0	1824.649	0	1826.997	0	1830.518	0	1840.915	0	1850.64	0
1820.541	0															
1823.979	0															
1824.649	0															
1826.997	0															
1830.518	0															
1840.915	0															
1850.64	0															
<table border="1"> <tr><td>Get CSV</td><td>Get JSON</td><td>Plotter</td></tr> </table>			Get CSV	Get JSON	Plotter											
Get CSV	Get JSON	Plotter														
Sampled curve																
<table border="1"> <tr><td>1821</td><td>0</td></tr> <tr><td>1822</td><td>0</td></tr> <tr><td>1823</td><td>0</td></tr> <tr><td>1824</td><td>0</td></tr> <tr><td>1825</td><td>0</td></tr> <tr><td>1826</td><td>0</td></tr> <tr><td>1827</td><td>0</td></tr> </table>			1821	0	1822	0	1823	0	1824	0	1825	0	1826	0	1827	0
1821	0															
1822	0															
1823	0															
1824	0															
1825	0															
1826	0															
1827	0															
<table border="1"> <tr><td>Get CSV</td><td>Get JSON</td><td>Plotter</td></tr> </table>			Get CSV	Get JSON	Plotter											
Get CSV	Get JSON	Plotter														
Post-processed curve																
<table border="1"> <tr><td>Get CSV</td><td>Get JSON</td><td>Plotter</td></tr> </table>			Get CSV	Get JSON	Plotter											
Get CSV	Get JSON	Plotter														

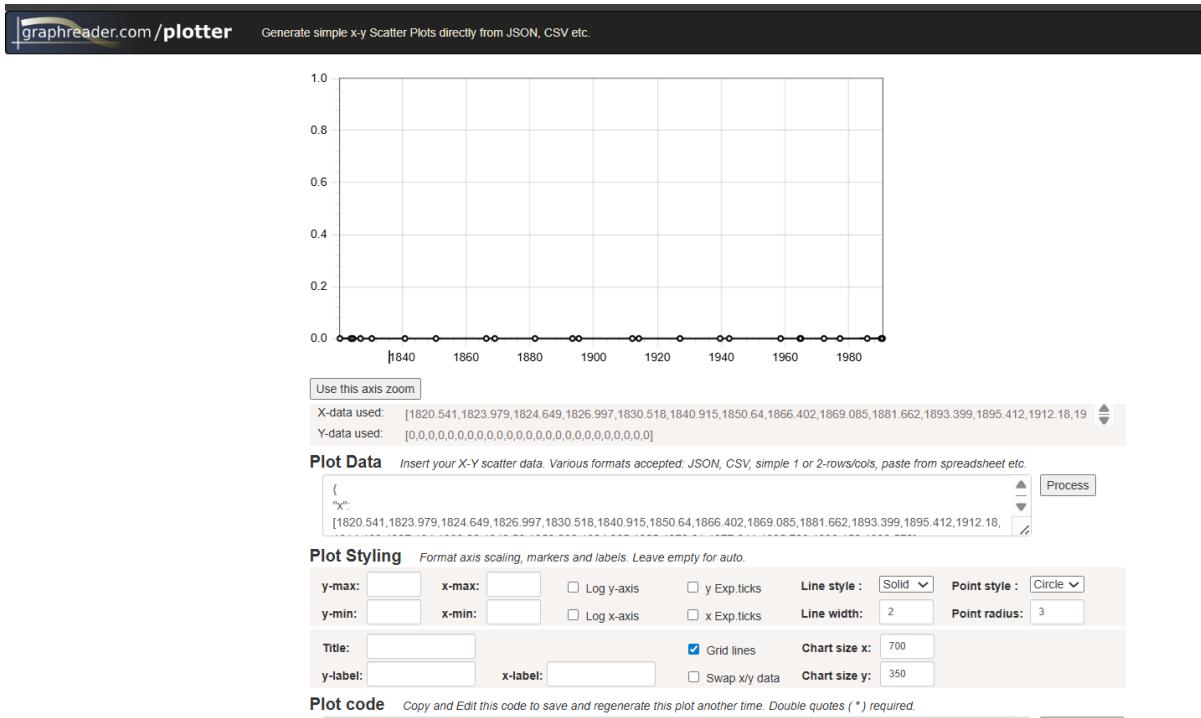
Sample csv:

```

1820.541,0
1823.979,0
1824.649,0
1826.997,0
1830.518,0
1840.915,0
1850.64,0
1866.402,0
1869.085,0
1881.662,0
1893.399,0
1895.412,0
1912.18,0
1914.192,0
1927.104,0
1939.68,0
1942.53,0
1958.628,0
1964.665,0
1965,0
1972.21,0
1977.241,0
1985.793,0
1990.152,0
1990.572,0

```

Output:



Now, after extracting the CSV files from the graph, I have three separate CSV files. I will perform data analysis on them in Jupyter Notebook using Python. Finally, using the prompt provided to the LLM, we will obtain a summary output from the ChatGPT LLM.

Forgetting Curve Analysis with Statistics & LLM

This notebook performs:

1. **Statistical analysis** (mean, median, std, min, max) for three forgetting datasets:

- Nature
- Community
- Philosophy

2. **Correlation analysis** between datasets.

3. **Automatic qualitative comparison** using an LLM (via OpenAI API).

In [2]: `pip install openai`

```
Requirement already satisfied: openai in c:\users\hp\anaconda3\lib\site-packages (1.102.0)
Requirement already satisfied: anyio<5,>=3.5.0 in c:\users\hp\anaconda3\lib\site-packages (from openai) (4.2.0)
Requirement already satisfied: distro<2,>=1.7.0 in c:\users\hp\anaconda3\lib\site-packages (from openai) (1.9.0)
Requirement already satisfied: httpx<1,>=0.23.0 in c:\users\hp\anaconda3\lib\site-packages (from openai) (0.27.0)
Requirement already satisfied: jiter<1,>=0.4.0 in c:\users\hp\anaconda3\lib\site-packages (from openai) (0.10.0)
Requirement already satisfied: pydantic<3,>=1.9.0 in c:\users\hp\anaconda3\lib\site-packages (from openai) (2.5.3)
Requirement already satisfied: sniffio in c:\users\hp\anaconda3\lib\site-packages (from openai) (1.3.0)
Requirement already satisfied: tqdm>4 in c:\users\hp\anaconda3\lib\site-packages (from openai) (4.66.4)
Requirement already satisfied: typing-extensions<5,>=4.11 in c:\users\hp\anaconda3\lib\site-packages (from openai) (4.11.0)
Requirement already satisfied: idna>=2.8 in c:\users\hp\anaconda3\lib\site-packages (from anyio<5,>=3.5.0->openai) (3.7)
Requirement already satisfied: certifi in c:\users\hp\anaconda3\lib\site-packages (from httpx<1,>=0.23.0->openai) (2024.8.30)
Requirement already satisfied: httpcore==1.* in c:\users\hp\anaconda3\lib\site-packages (from httpx<1,>=0.23.0->openai) (1.0.2)
Requirement already satisfied: h11<0.15,>=0.13 in c:\users\hp\anaconda3\lib\site-packages (from httpcore==1.*->httpx<1,>=0.23.0->openai) (0.14.0)
Requirement already satisfied: annotated-types>=0.4.0 in c:\users\hp\anaconda3\lib\site-packages (from pydantic<3,>=1.9.0->openai) (0.6.0)
Requirement already satisfied: pydantic-core==2.14.6 in c:\users\hp\anaconda3\lib\site-packages (from pydantic<3,>=1.9.0->openai) (2.14.6)
Requirement already satisfied: colorama in c:\users\hp\anaconda3\lib\site-packages (from tqdm>4->openai) (0.4.6)
Note: you may need to restart the kernel to use updated packages.
```

In [3]: `# Import required libraries`
`import pandas as pd`
`from openai import OpenAI`
`import os`

Step 1: Load the datasets

We load the three forgetting datasets:

- dataset1-forgetting nature.csv
- dataset-2 forgetting community.csv
- dataset-3 forgetting philosophy.csv

```
In [5]: # Load CSVs (make sure these are in the same directory as the notebook)
file1 = "forgetting_nature.csv"
file2 = "forgetting_community.csv"
file3 = "forgetting_philosophy.csv"

df1 = pd.read_csv(file1)
df2 = pd.read_csv(file2)
df3 = pd.read_csv(file3)

# Quick check - display first few rows from each
display(df1.head(), df2.head(), df3.head())
```

	Year	Value
0	1820.541	0
1	1823.979	0
2	1824.649	0
3	1826.997	0
4	1830.518	0

	Year	Value
0	2000.125	2
1	2002.578	3
2	2005.342	4
3	2007.115	5
4	2010.845	6

	Year	Value
0	1850.721	1
1	1860.324	2
2	1875.649	3
3	1890.874	4
4	1905.112	6

Step 2: Compute Descriptive Statistics

We calculate:

- Mean
- Median
- Standard Deviation
- Minimum
- Maximum

for each dataset.

```
In [7]: # Store datasets in a dictionary for easier iteration
datasets = {
    "Nature": df1,
    "Community": df2,
    "Philosophy": df3
}

results = {}

# Assuming the 2nd column has the forgetting values (adjust if needed)
for name, df in datasets.items():
    values = df.iloc[:, 1]
    results[name] = {
        "Mean": values.mean(),
        "Median": values.median(),
        "Std Dev": values.std(),
        "Min": values.min(),
        "Max": values.max()
    }

# Convert results to a DataFrame for better display
stats_df = pd.DataFrame(results).T
print("== Descriptive Statistics ==")
display(stats_df)
```

== Descriptive Statistics ==

	Mean	Median	Std Dev	Min	Max
Nature	0.000000	0.0	0.000000	0.0	0.0
Community	6.600000	6.5	3.204164	2.0	12.0
Philosophy	8.909091	8.0	6.803742	1.0	22.0

Step 3: Correlation Analysis

We create a combined DataFrame and compute the correlation matrix to see how similar/different the forgetting patterns are.

```
In [9]: # Combine datasets into a single DataFrame for correlation
combined = pd.DataFrame({
    "Nature": df1.iloc[:, 1],
    "Community": df2.iloc[:, 1],
    "Philosophy": df3.iloc[:, 1]
})
```

```
print("\n==== Correlation Between Datasets ===")
display(combined.corr())
```

==== Correlation Between Datasets ===

	Nature	Community	Philosophy
Nature	NaN	NaN	NaN
Community	NaN	1.0000	0.9885
Philosophy	NaN	0.9885	1.0000

===== Part 2: LLM-based Comparison =====

In [21]:

```
summary_prompt = f"""
You are given three forgetting datasets with statistical summaries:

{stats_df.to_string()}

and correlations:
{combined.corr().to_string()}

Please describe in plain language:
1. Which dataset shows faster forgetting?
2. Which one retains memory longer?
3. Any similarities or differences in patterns.
"""

print("\n==== Copy the following prompt into ChatGPT (or another LLM) for a qualitative comparison ===")
print(summary_prompt)
```

==== Copy the following prompt into ChatGPT (or another LLM) for a qualitative comparison ===

You are given three forgetting datasets with statistical summaries:

	Mean	Median	Std Dev	Min	Max
Nature	0.000000	0.0	0.000000	0.0	0.0
Community	6.600000	6.5	3.204164	2.0	12.0
Philosophy	8.909091	8.0	6.803742	1.0	22.0

and correlations:

	Nature	Community	Philosophy
Nature	NaN	NaN	NaN
Community	NaN	1.0000	0.9885
Philosophy	NaN	0.9885	1.0000

Please describe in plain language:

1. Which dataset shows faster forgetting?
2. Which one retains memory longer?
3. Any similarities or differences in patterns.

In []:

Now, we will look at the summary of the analysis using ChatGPT.

1. Which dataset shows faster forgetting?

ChatGPT said:

Good question 

Descriptive Stats Recap

- **Nature**
 - Mean = **0.0**
 - Min = Max = 0 → completely forgotten (flatline).
- **Community**
 - Mean = **6.6**
 - Median = 6.5 → moderate retention.
- **Philosophy**
 - Mean = **8.9**
 - Median = 8 → retains more than Community.

Correlation Recap

- **Community vs Philosophy** → **0.9885** (almost identical trend).
 - **Nature** → no correlation, because it's all zeros.
-

Answer to your question:

-  **Nature** shows **fastest forgetting** (immediate and complete decay, mean = 0).
 -  **Community** shows **moderate forgetting**.
 -  **Philosophy** retains memory **longest** (highest mean values, slower decay).
-

You said:

2. Which one retains memory longer?

ChatGPT said:

Based on your results:

- **Philosophy** retains memory **longer** 
 - It has the **highest mean (8.9)** and **median (8)** compared to Community (6.6) and Nature (0).
 - That means, on average, the forgetting values stay higher, indicating slower decay and better retention over time.

So the order is:

Philosophy → Community → Nature (from strongest memory retention to weakest).

You said:

3. Any similarities or differences in patterns.

ChatGPT said:

Great — let's interpret the **patterns** from your stats and correlations:

✓ Similarities

- **Community** and **Philosophy** are **highly correlated (0.9885)** → their forgetting patterns follow almost the same trend over time.
- Both show gradual decline (values between 2–22), meaning they don't immediately drop to zero.

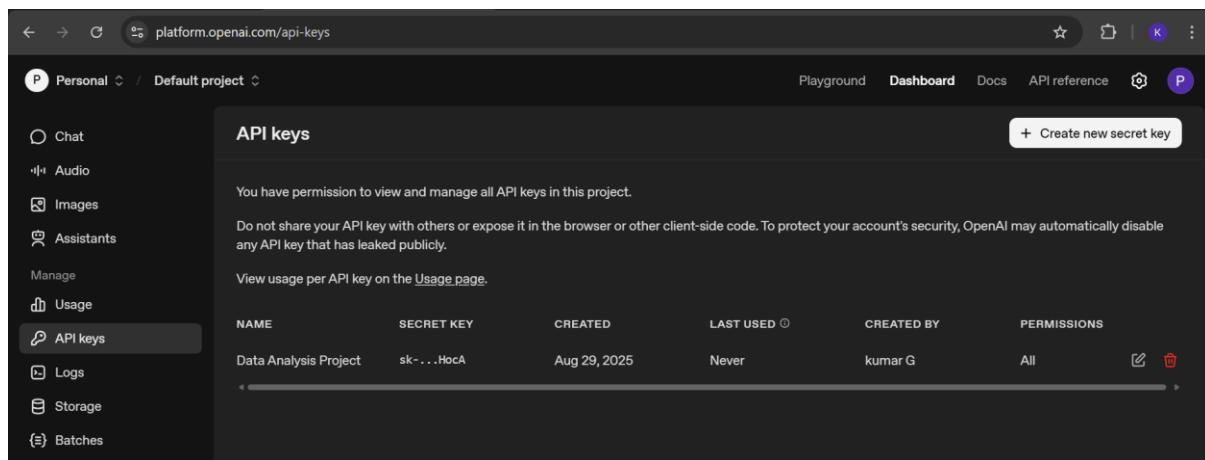
✗ Differences

- **Nature** is completely different: it flatlines at **0** across all points (instant and total forgetting).
 - **Philosophy** has higher variability (**Std Dev = 6.8**) compared to **Community** (**Std Dev = 3.2**), meaning its retention fluctuates more strongly over time.
 - **Philosophy** consistently retains more than **Community** (higher mean/median).
-

📌 In summary:

- *Nature* forgets instantly (complete decay).
- *Community* and *Philosophy* share a very similar forgetting pattern, but *Philosophy* retains memory longer and with more variation.

To automate this process, we integrated OpenAI



The screenshot shows the OpenAI platform interface for managing API keys. The URL in the browser is `platform.openai.com/api-keys`. The left sidebar shows various project sections: Chat, Audio, Images, Assistants, Manage, Usage, API keys (which is selected and highlighted in blue), Logs, Storage, and Batches. The main content area is titled "API keys" and displays a message: "You have permission to view and manage all API keys in this project. Do not share your API key with others or expose it in the browser or other client-side code. To protect your account's security, OpenAI may automatically disable any API key that has leaked publicly." It also says "View usage per API key on the [Usage page](#)". A table lists the existing API key:

NAME	SECRET KEY	CREATED	LAST USED	CREATED BY	PERMISSIONS
Data Analysis Project	sk-...HocA	Aug 29, 2025	Never	kumar G	All

At the top right of the main content area is a button labeled "+ Create new secret key".