Modern NLP Approach - Topic Extraction

Load The Dataset

```
# Mount Google Drive (if using Colab)
from google.colab import drive
drive.mount('/content/drive')

import pandas as pd

# Load cleaned dataset
input_path = "/content/drive/My Drive/NLP/Assignment_3/fomc_transcripts_spacy_cleaned.csv
df = pd.read_csv(input_path)
df['Date'] = pd.to_datetime(df['Date'])
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
```

```
df
```

	URL	Date	Year	Month	Day	Content	date
0	https://www.federalreserve.gov/monetarypolicy/...	2025-03-19	2025	2025-03-01	19	HomeMonetary PolicyFederal Open Market Committ...	2025-03-31
1	https://www.federalreserve.gov/monetarypolicy/...	2025-01-29	2025	2025-01-01	29	HomeMonetary PolicyFederal Open Market Committ...	2025-01-31
2	https://www.federalreserve.gov/monetarypolicy/...	2024-12-18	2024	2024-12-01	18	HomeMonetary PolicyFederal Open Market Committ...	2024-12-31
3	https://www.federalreserve.gov/monetarypolicy/...	2024-11-07	2024	2024-11-01	7	HomeMonetary PolicyFederal Open Market Committ...	2024-11-30
4	https://www.federalreserve.gov/monetarypolicy/...	2024-09-18	2024	2024-09-01	18	HomeMonetary PolicyFederal Open Market Committ...	2024-09-30
...	...	...	...	...	...	...	...

61	https://www.federalreserve.gov/monetarypolicy/...	2015-07-29	2015	2015-07-01	29	HomeMonetary PolicyFederal Open Market Committ...	2015-07-31
62	https://www.federalreserve.gov/monetarypolicy/...	2015-06-17	2015	2015-06-01	17	HomeMonetary PolicyFederal Open Market Committ...	2015-06-30
63	https://www.federalreserve.gov/monetarypolicy/...	2015-04-29	2015	2015-04-01	29	HomeMonetary PolicyFederal Open Market Committ...	2015-04-30
64	https://www.federalreserve.gov/monetarypolicy/...	2015-03-18	2015	2015-03-01	18	HomeMonetary PolicyFederal Open Market Committ...	2015-03-31
65	https://www.federalreserve.gov/monetarypolicy/...	2015-01-28	2015	2015-01-01	28	HomeMonetary PolicyFederal Open Market Committ...	2015-01-31

66 rows × 20 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
import pandas as pd

# 1) Convert the 'Date' column from object (string) to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# 2) (Optional) Drop or inspect rows where conversion failed
# print(df[df['Date'].isna()])

# 3) Sort the DataFrame by the new datetime 'Date' column in descending order
df = df.sort_values(by='Date', ascending=False)

# 4) (Optional) Reset index if you want a clean integer index
df = df.reset_index(drop=True)

# Now df['Date'] is a datetime dtype and the rows are sorted newest → oldest
#print(df[['doc_id', 'Date']].head())


# 1) Make sure your index is reset 0…N-1
df = df.reset_index(drop=True)

# 2) Rebuild doc_id as "doc1", "doc2", … up to "docN"
```

```
df['doc_id'] = 'doc' + (df.index + 1).astype(str)

# 3) Check
print(df[['doc_id', 'Date']].head())
```

```
  doc_id       Date
0   doc1 2025-03-19
1   doc2 2025-01-29
2   doc3 2024-12-18
3   doc4 2024-11-07
4   doc5 2024-09-18
```

```
df
```

	URL	Date	Year	Month	Day	Content	date
0	https://www.federalreserve.gov/monetarypolicy/...	2025-03-19	2025	2025-03-01	19	HomeMonetary PolicyFederal Open Market Committ...	2025-03-31
1	https://www.federalreserve.gov/monetarypolicy/...	2025-01-29	2025	2025-01-01	29	HomeMonetary PolicyFederal Open Market Committ...	2025-01-31
2	https://www.federalreserve.gov/monetarypolicy/...	2024-12-18	2024	2024-12-01	18	HomeMonetary PolicyFederal Open Market Committ...	2024-12-31
3	https://www.federalreserve.gov/monetarypolicy/...	2024-11-07	2024	2024-11-01	7	HomeMonetary PolicyFederal Open Market Committ...	2024-11-30
4	https://www.federalreserve.gov/monetarypolicy/...	2024-09-18	2024	2024-09-01	18	HomeMonetary PolicyFederal Open Market Committ...	2024-09-30
...	...	...	...	...	...	...	...
61	https://www.federalreserve.gov/monetarypolicy/...	2015-07-29	2015	2015-07-01	29	HomeMonetary PolicyFederal Open Market Committ...	2015-07-31
62	https://www.federalreserve.gov/monetarypolicy/...	2015-06-17	2015	2015-06-01	17	HomeMonetary PolicyFederal Open Market Committ...	2015-06-30
63	https://www.federalreserve.gov/	2015-04-29	2015	2015-04-01	29	HomeMonetary PolicyFederal	2015-04-30

	monetarypolicy/...					Open Market Committ...	
64	https://www.federalreserve.gov/monetarypolicy/...	2015-03-18	2015	2015-03-01	18	HomeMonetary PolicyFederal Open Market Committ...	2015-03-31
65	https://www.federalreserve.gov/monetarypolicy/...	2015-01-28	2015	2015-01-01	28	HomeMonetary PolicyFederal Open Market Committ...	2015-01-31

66 rows × 21 columns

```
!pip install --upgrade transformers
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.11/dist-package
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (f
Requirement already satisfied: huggingface-hub<1.0,>=0.30.0 in /usr/local/lib/python3
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-pack
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.11/dist-pa
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (f
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.11/di
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.11/dist-pac
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.1
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-package
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-p
```

```
# 🚨 Force reinstall PyTorch & NumPy without cache to fix compatibility
!pip install --force-reinstall --no-cache-dir torch==2.1.0 torchvision==0.16.0 numpy==1.2
```

```
Collecting torch==2.1.0
  Downloading torch-2.1.0-cp311-cp311-manylinux1_x86_64.whl.metadata (25 kB)
Collecting torchvision==0.16.0
  Downloading torchvision-0.16.0-cp311-cp311-manylinux1_x86_64.whl.metadata (6.6 kB)
Collecting numpy==1.26.4
  Downloading numpy-1.26.4-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 61.0/61.0 kB 6.8 MB/s eta 0:00:00
Collecting filelock (from torch==2.1.0)
  Downloading filelock-3.18.0-py3-none-any.whl.metadata (2.9 kB)
Collecting typing-extensions (from torch==2.1.0)
  Downloading typing_extensions-4.13.2-py3-none-any.whl.metadata (3.0 kB)
Collecting sympy (from torch==2.1.0)
  Downloading sympy-1.14.0-py3-none-any.whl.metadata (12 kB)
```

```
Collecting networkx (from torch==2.1.0)
  Downloading networkx-3.4.2-py3-none-any.whl.metadata (6.3 kB)
Collecting jinja2 (from torch==2.1.0)
  Downloading jinja2-3.1.6-py3-none-any.whl.metadata (2.9 kB)
Collecting fsspec (from torch==2.1.0)
  Downloading fsspec-2025.3.2-py3-none-any.whl.metadata (11 kB)
Collecting nvidia-cuda-nvrtc-cu12==12.1.105 (from torch==2.1.0)
  Downloading nvidia_cuda_nvrtc_cu12-12.1.105-py3-none-manylinux1_x86_64.whl.metadata
Collecting nvidia-cuda-runtime-cu12==12.1.105 (from torch==2.1.0)
  Downloading nvidia_cuda_runtime_cu12-12.1.105-py3-none-manylinux1_x86_64.whl.metada
Collecting nvidia-cuda-cupti-cu12==12.1.105 (from torch==2.1.0)
  Downloading nvidia_cuda_cupti_cu12-12.1.105-py3-none-manylinux1_x86_64.whl.metadata
Collecting nvidia-cudnn-cu12==8.9.2.26 (from torch==2.1.0)
  Downloading nvidia_cudnn_cu12-8.9.2.26-py3-none-manylinux1_x86_64.whl.metadata (1.6
Collecting nvidia-cublas-cu12==12.1.3.1 (from torch==2.1.0)
  Downloading nvidia_cublas_cu12-12.1.3.1-py3-none-manylinux1_x86_64.whl.metadata (1.
Collecting nvidia-cufft-cu12==11.0.2.54 (from torch==2.1.0)
  Downloading nvidia_cufft_cu12-11.0.2.54-py3-none-manylinux1_x86_64.whl.metadata (1.
Collecting nvidia-curand-cu12==10.3.2.106 (from torch==2.1.0)
  Downloading nvidia_curand_cu12-10.3.2.106-py3-none-manylinux1_x86_64.whl.metadata (
Collecting nvidia-cusolver-cu12==11.4.5.107 (from torch==2.1.0)
  Downloading nvidia_cusolver_cu12-11.4.5.107-py3-none-manylinux1_x86_64.whl.metadata
Collecting nvidia-cusparse-cu12==12.1.0.106 (from torch==2.1.0)
  Downloading nvidia_cusparse_cu12-12.1.0.106-py3-none-manylinux1_x86_64.whl.metadata
Collecting nvidia-nccl-cu12==2.18.1 (from torch==2.1.0)
  Downloading nvidia_nccl_cu12-2.18.1-py3-none-manylinux1_x86_64.whl.metadata (1.8 kB
Collecting nvidia-nvtx-cu12==12.1.105 (from torch==2.1.0)
  Downloading nvidia_nvtx_cu12-12.1.105-py3-none-manylinux1_x86_64.whl.metadata (1.7
Collecting triton==2.1.0 (from torch==2.1.0)
  Downloading triton-2.1.0-0-cp311-cp311-manylinux2014_x86_64.manylinux_2_17_x86_64.w
Collecting requests (from torchvision==0.16.0)
  Downloading requests-2.32.3-py3-none-any.whl.metadata (4.6 kB)
Collecting pillow!=8.3.*,>=5.3.0 (from torchvision==0.16.0)
  Downloading pillow-11.2.1-cp311-cp311-manylinux_2_28_x86_64.whl.metadata (8.9 kB)
Collecting nvidia-nvjitlink-cu12 (from nvidia-cusolver-cu12==11.4.5.107->torch==2.1.0
  Downloading nvidia_nvjitlink_cu12-12.9.41-py3-none-manylinux2010_x86_64.manylinux_2
Collecting MarkupSafe>=2.0 (from jinja2->torch==2.1.0)
  Downloading MarkupSafe-3.0.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64
Collecting charset-normalizer<4,>=2 (from requests->torchvision==0.16.0)
  Downloading charset_normalizer-3.4.2-cp311-cp311-manylinux_2_17_x86_64.manylinux201
Collecting idna<4,>=2.5 (from requests->torchvision==0.16.0)
  Downloading idna-3.10-py3-none-any.whl.metadata (10 kB)
Collecting urllib3<3,>=1.21.1 (from requests->torchvision==0.16.0)
  Downloading urllib3-2.4.0-py3-none-any.whl.metadata (6.5 kB)
Collecting certifi>=2017.4.17 (from requests->torchvision==0.16.0)
  Downloading certifi-2025.4.26-py3-none-any.whl.metadata (2.5 kB)
Collecting mpmath<1.4,>=1.1.0 (from sympy->torch==2.1.0)
  Downloading mpmath-1.3.0-py3-none-any.whl.metadata (8.6 kB)
Downloading torch-2.1.0-cp311-cp311-manylinux1_x86_64.whl (670.2 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 670.2/670.2 MB 138.6 MB/s eta 0:00:00
Downloading torchvision-0.16.0-cp311-cp311-manylinux1_x86_64.whl (6.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.9/6.9 MB 191.2 MB/s eta 0:00:00
Downloading numpy-1.26.4-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 18.3/18.3 MB 209.1 MB/s eta 0:00:00
Downloading nvidia_cublas_cu12-12.1.3.1-py3-none-manylinux1_x86_64.whl (410.6 MB)
```

```
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 410.6/410.6 MB 232.6 MB/s eta 0:00:00
Downloading nvidia_cuda_cupti_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (14.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 14.1/14.1 MB 247.3 MB/s eta 0:00:00
Downloading nvidia_cuda_nvrtc_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (23.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 23.7/23.7 MB 126.1 MB/s eta 0:00:00
Downloading nvidia_cuda_runtime_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (823 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 823.6/823.6 kB 247.8 MB/s eta 0:00:00
Downloading nvidia_cudnn_cu12-8.9.2.26-py3-none-manylinux1_x86_64.whl (731.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 731.7/731.7 MB 172.0 MB/s eta 0:00:00
Downloading nvidia_cufft_cu12-11.0.2.54-py3-none-manylinux1_x86_64.whl (121.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 121.6/121.6 MB 145.9 MB/s eta 0:00:00
Downloading nvidia_curand_cu12-10.3.2.106-py3-none-manylinux1_x86_64.whl (56.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.5/56.5 MB 144.3 MB/s eta 0:00:00
Downloading nvidia_cusolver_cu12-11.4.5.107-py3-none-manylinux1_x86_64.whl (124.2 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 124.2/124.2 MB 164.3 MB/s eta 0:00:00
Downloading nvidia_cusparse_cu12-12.1.0.106-py3-none-manylinux1_x86_64.whl (196.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 196.0/196.0 MB 238.2 MB/s eta 0:00:00
Downloading nvidia_nccl_cu12-2.18.1-py3-none-manylinux1_x86_64.whl (209.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 209.8/209.8 MB 207.7 MB/s eta 0:00:00
Downloading nvidia_nvtx_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (99 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 99.1/99.1 kB 263.6 MB/s eta 0:00:00
Downloading triton-2.1.0-0-cp311-cp311-manylinux2014_x86_64.manylinux_2_17_x86_64.whl
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 89.2/89.2 MB 213.3 MB/s eta 0:00:00
Downloading pillow-11.2.1-cp311-cp311-manylinux_2_28_x86_64.whl (4.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.6/4.6 MB 197.8 MB/s eta 0:00:00
Downloading filelock-3.18.0-py3-none-any.whl (16 kB)
Downloading fsspec-2025.3.2-py3-none-any.whl (194 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.4/194.4 kB 247.8 MB/s eta 0:00:00
Downloading jinja2-3.1.6-py3-none-any.whl (134 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 134.9/134.9 kB 202.5 MB/s eta 0:00:00
Downloading networkx-3.4.2-py3-none-any.whl (1.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.7/1.7 MB 271.5 MB/s eta 0:00:00
Downloading requests-2.32.3-py3-none-any.whl (64 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 64.9/64.9 kB 124.0 MB/s eta 0:00:00
Downloading sympy-1.14.0-py3-none-any.whl (6.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.3/6.3 MB 242.6 MB/s eta 0:00:00
Downloading typing_extensions-4.13.2-py3-none-any.whl (45 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 45.8/45.8 kB 175.6 MB/s eta 0:00:00
Downloading certifi-2025.4.26-py3-none-any.whl (159 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 159.6/159.6 kB 257.7 MB/s eta 0:00:00
Downloading charset_normalizer-3.4.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 147.3/147.3 kB 267.4 MB/s eta 0:00:00
Downloading idna-3.10-py3-none-any.whl (70 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 70.4/70.4 kB 244.5 MB/s eta 0:00:00
Downloading MarkupSafe-3.0.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.w
Downloading mpmath-1.3.0-py3-none-any.whl (536 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 536.2/536.2 kB 112.6 MB/s eta 0:00:00
Downloading urllib3-2.4.0-py3-none-any.whl (128 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 128.7/128.7 kB 282.7 MB/s eta 0:00:00
Downloading nvidia_nvjitlink_cu12-12.9.41-py3-none-manylinux2010_x86_64.manylinux_2_1
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 39.7/39.7 MB 198.9 MB/s eta 0:00:00
Installing collected packages: mpmath, urllib3, typing-extensions, sympy, pillow, nvi
  Attempting uninstall: mpmath
    Found existing installation: mpmath 1.3.0
    Uninstalling mpmath-1.3.0:
      Successfully uninstalled mpmath-1.3.0
  Attempting uninstall: urllib3
    Found existing installation: urllib3 2.4.0
    Uninstalling urllib3-2.4.0:
      Successfully uninstalled urllib3-2.4.0
  Attempting uninstall: typing-extensions
    Found existing installation: typing_extensions 4.13.2
    Uninstalling typing_extensions-4.13.2:
      Successfully uninstalled typing_extensions-4.13.2
  Attempting uninstall: sympy
    Found existing installation: sympy 1.14.0
    Uninstalling sympy-1.14.0:
      Successfully uninstalled sympy-1.14.0
  Attempting uninstall: pillow
    Found existing installation: pillow 11.2.1
    Uninstalling pillow-11.2.1:
      Successfully uninstalled pillow-11.2.1
  Attempting uninstall: nvidia-nvtx-cu12
    Found existing installation: nvidia-nvtx-cu12 12.6.77
    Uninstalling nvidia-nvtx-cu12-12.6.77:
      Successfully uninstalled nvidia-nvtx-cu12-12.6.77
  Attempting uninstall: nvidia-nvjitlink-cu12
    Found existing installation: nvidia-nvjitlink-cu12 12.6.85
    Uninstalling nvidia-nvjitlink-cu12-12.6.85:
      Successfully uninstalled nvidia-nvjitlink-cu12-12.6.85
  Attempting uninstall: nvidia-nccl-cu12
    Found existing installation: nvidia-nccl-cu12 2.26.2
    Uninstalling nvidia-nccl-cu12-2.26.2:
      Successfully uninstalled nvidia-nccl-cu12-2.26.2
  Attempting uninstall: nvidia-curand-cu12
    Found existing installation: nvidia-curand-cu12 10.3.7.77
    Uninstalling nvidia-curand-cu12-10.3.7.77:
      Successfully uninstalled nvidia-curand-cu12-10.3.7.77
  Attempting uninstall: nvidia-cufft-cu12
    Found existing installation: nvidia-cufft-cu12 11.3.0.4
    Uninstalling nvidia-cufft-cu12-11.3.0.4:
      Successfully uninstalled nvidia-cufft-cu12-11.3.0.4
  Attempting uninstall: nvidia-cuda-runtime-cu12
    Found existing installation: nvidia-cuda-runtime-cu12 12.6.77
    Uninstalling nvidia-cuda-runtime-cu12-12.6.77:
      Successfully uninstalled nvidia-cuda-runtime-cu12-12.6.77
  Attempting uninstall: nvidia-cuda-nvrtc-cu12
    Found existing installation: nvidia-cuda-nvrtc-cu12 12.6.77
    Uninstalling nvidia-cuda-nvrtc-cu12-12.6.77:
      Successfully uninstalled nvidia-cuda-nvrtc-cu12-12.6.77
  Attempting uninstall: nvidia-cuda-cupti-cu12
    Found existing installation: nvidia-cuda-cupti-cu12 12.6.80
    Uninstalling nvidia-cuda-cupti-cu12-12.6.80:
      Successfully uninstalled nvidia-cuda-cupti-cu12-12.6.80
  Attempting uninstall: nvidia-cublas-cu12
    Found existing installation: nvidia-cublas-cu12 12.6.4.1
    Uninstalling nvidia-cublas-cu12-12.6.4.1:
      Successfully uninstalled nvidia-cublas-cu12-12.6.4.1
```

```
Attempting uninstall: numpy
  Found existing installation: numpy 2.0.2
  Uninstalling numpy-2.0.2:
    Successfully uninstalled numpy-2.0.2
Attempting uninstall: networkx
  Found existing installation: networkx 3.4.2
  Uninstalling networkx-3.4.2:
    Successfully uninstalled networkx-3.4.2
Attempting uninstall: MarkupSafe
  Found existing installation: MarkupSafe 3.0.2
  Uninstalling MarkupSafe-3.0.2:
    Successfully uninstalled MarkupSafe-3.0.2
Attempting uninstall: idna
  Found existing installation: idna 3.10
  Uninstalling idna-3.10:
    Successfully uninstalled idna-3.10
Attempting uninstall: fsspec
  Found existing installation: fsspec 2025.3.2
  Uninstalling fsspec-2025.3.2:
    Successfully uninstalled fsspec-2025.3.2
Attempting uninstall: filelock
  Found existing installation: filelock 3.18.0
  Uninstalling filelock-3.18.0:
    Successfully uninstalled filelock-3.18.0
Attempting uninstall: charset-normalizer
  Found existing installation: charset-normalizer 3.4.1
  Uninstalling charset-normalizer-3.4.1:
    Successfully uninstalled charset-normalizer-3.4.1
Attempting uninstall: certifi
  Found existing installation: certifi 2025.4.26
  Uninstalling certifi-2025.4.26:
    Successfully uninstalled certifi-2025.4.26
Attempting uninstall: triton
  Found existing installation: triton 3.3.0
  Uninstalling triton-3.3.0:
    Successfully uninstalled triton-3.3.0
Attempting uninstall: requests
  Found existing installation: requests 2.32.3
  Uninstalling requests-2.32.3:
    Successfully uninstalled requests-2.32.3
Attempting uninstall: nvidia-cusparse-cu12
  Found existing installation: nvidia-cusparse-cu12 12.5.4.2
  Uninstalling nvidia-cusparse-cu12-12.5.4.2:
    Successfully uninstalled nvidia-cusparse-cu12-12.5.4.2
Attempting uninstall: nvidia-cudnn-cu12
  Found existing installation: nvidia-cudnn-cu12 9.5.1.17
  Uninstalling nvidia-cudnn-cu12-9.5.1.17:
    Successfully uninstalled nvidia-cudnn-cu12-9.5.1.17
Attempting uninstall: jinja2
  Found existing installation: Jinja2 3.1.6
  Uninstalling Jinja2-3.1.6:
    Successfully uninstalled Jinja2-3.1.6
Attempting uninstall: nvidia-cusolver-cu12
  Found existing installation: nvidia-cusolver-cu12 11.7.1.2
```

```
    Uninstalling nvidia-cusolver-cu12-11.7.1.2:
      Successfully uninstalled nvidia-cusolver-cu12-11.7.1.2
  Attempting uninstall: torch
    Found existing installation: torch 2.7.0
    Uninstalling torch-2.7.0:
      Successfully uninstalled torch-2.7.0
  Attempting uninstall: torchvision
    Found existing installation: torchvision 0.21.0+cu124
    Uninstalling torchvision-0.21.0+cu124:
      Successfully uninstalled torchvision-0.21.0+cu124
ERROR: pip's dependency resolver does not currently take into account all the package
torchaudio 2.6.0+cu124 requires torch==2.6.0, but you have torch 2.1.0 which is incom
thinc 8.3.6 requires numpy<3.0.0,>=2.0.0, but you have numpy 1.26.4 which is incompat
Successfully installed MarkupSafe-3.0.2 certifi-2025.4.26 charset-normalizer-3.4.2 fi
WARNING: The following packages were previously imported in this runtime:
  [PIL,certifi,charset_normalizer,markupsafe,requests,torch,torchgen]
You must restart the runtime in order to use newly installed versions.
```

RESTART SESSION

```
!pip install --upgrade spacy
```

```
Requirement already satisfied: spacy in /usr/local/lib/python3.11/dist-packages (3.8.
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.11
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.11/dist-
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.11/dis
Requirement already satisfied: thinc<8.4.0,>=8.3.4 in /usr/local/lib/python3.11/dist-
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.11/dist
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.11/dist-
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.11/d
Requirement already satisfied: weasel<0.5.0,>=0.1.0 in /usr/local/lib/python3.11/dist
Requirement already satisfied: typer<1.0.0,>=0.3.0 in /usr/local/lib/python3.11/dist-
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.11/dist-
Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.11/dist-packag
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/d
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages (fro
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-pack
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.11/d
Requirement already satisfied: language-data>=1.2 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/di
Requirement already satisfied: pydantic-core==2.33.1 in /usr/local/lib/python3.11/dis
Requirement already satisfied: typing-extensions>=4.12.2 in /usr/local/lib/python3.11
Requirement already satisfied: typing-inspection>=0.4.0 in /usr/local/lib/python3.11/
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-package
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: blis<1.4.0,>=1.3.0 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.11/
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.11/
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.11/dist-package
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.11/dist-packag
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/python3.1
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/python3.11/
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/dist-pack
Requirement already satisfied: marisa-trie>=1.1.0 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dis
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/d
Requirement already satisfied: wrapt in /usr/local/lib/python3.11/dist-packages (from
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages
```

```
import spacy
```

```
nlp = spacy.load("en_core_web_sm")
```

```
import torch
from transformers import AutoTokenizer, AutoModel
import spacy

# Load spaCy for sentence segmentation
nlp = spacy.load("en_core_web_sm")

# Load FinBERT-tone tokenizer & model
tokenizer = AutoTokenizer.from_pretrained("yiyanghkust/finbert-tone")
model     = AutoModel.from_pretrained("yiyanghkust/finbert-tone")
model.eval()
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)
```

```
pytorch_model.bin: 100%                                   439M/439M [00:02<00:00, 169MB/
                                                                                       s]
/usr/local/lib/python3.11/dist-packages/torch/_utils.py:831: UserWarning: TypedStorag
  return self.fget.__get__(instance, owner)()
model.safetensors:   0%                                   0.00/439M [00:00<?, ?B/s]
BertModel(
  (embeddings): BertEmbeddings(
    (word_embeddings): Embedding(30873, 768, padding_idx=0)
    (position_embeddings): Embedding(512, 768)
    (token_type_embeddings): Embedding(2, 768)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
  (encoder): BertEncoder(
    (layer): ModuleList(
      (0-11): 12 x BertLayer(
        (attention): BertAttention(
          (self): BertSelfAttention(
            (query): Linear(in_features=768, out_features=768, bias=True)
            (key): Linear(in_features=768, out_features=768, bias=True)
            (value): Linear(in_features=768, out_features=768, bias=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
          (output): BertSelfOutput(
            (dense): Linear(in_features=768, out_features=768, bias=True)
            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
        (intermediate): BertIntermediate(
          (dense): Linear(in_features=768, out_features=3072, bias=True)
          (intermediate_act_fn): GELUActivation()
        )
        (output): BertOutput(
          (dense): Linear(in_features=3072, out_features=768, bias=True)
          (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
          (dropout): Dropout(p=0.1, inplace=False)
        )
      )
    )
  )
  (pooler): BertPooler(
    (dense): Linear(in_features=768, out_features=768, bias=True)
    (activation): Tanh()
  )
)
```

```
# ───────────────────────────────────────────────────────────────────────────────
# 1. IMPORTS, NLP & CHUNKING STRATEGY (450-TOKEN, SENTENCE-PRESERVED)
# ───────────────────────────────────────────────────────────────────────────────


def chunk_document(text: str,
                   max_tokens: int = 450,
                   overlap: int = 0
                  ) -> list[str]:
    """
    Split `text` into chunks of ≤ max_tokens (by tokenizer count),
    preserving whole sentences. Overlap = number of tokens to
    re-use between consecutive chunks.
    """
    sentences = [sent.text.strip() for sent in nlp(text).sents]
    chunks, current, current_len = [], [], 0

    for sent in sentences:
        sent_len = len(tokenizer.tokenize(sent))
        if current_len + sent_len > max_tokens:
            chunks.append(" ".join(current))
            if overlap > 0 and current:
                # keep last `overlap` tokens as starting point
                ov_tokens = tokenizer.tokenize(" ".join(current))[-overlap:]
                ov_text = tokenizer.convert_tokens_to_string(ov_tokens)
                current = [ov_text]
                current_len = len(ov_tokens)
            else:
                current, current_len = [], 0
        current.append(sent)
        current_len += sent_len

    if current:
        chunks.append(" ".join(current))
    return chunks

def embed_texts(texts: list[str], max_length: int = 450) -> torch.Tensor:
    """
    Embed each text chunk by mean-pooling token embeddings.
    `max_length` ensures we pad/truncate to the chunk size.
    Returns a (n_chunks × hidden_size) tensor.
    """
    embeddings = []
    with torch.no_grad():
        for txt in texts:
            inputs = tokenizer(
                txt,
                padding="max_length",
                truncation=True,
                max_length=max_length,
                return_tensors="pt"
            ).to(device)
            outputs = model(**inputs).last_hidden_state  # [1, L, H]
            mask = inputs["attention_mask"].unsqueeze(-1)  # [1, L, 1]
            summed = (outputs * mask).sum(dim=1)
            counts = mask.sum(dim=1)
            embeddings.append((summed / counts).squeeze(0))
    return torch.stack(embeddings)

# === USAGE ===
# 1. Break each document into 450-token chunks
docs = df["cleaned_text"].tolist()
docs_chunks = [chunk_document(doc, max_tokens=450, overlap=0) for doc in docs]

# 2. Flatten for embedding
all_chunks = [chunk for sublist in docs_chunks for chunk in sublist]

# 3. Compute embeddings on 450-token chunks
chunk_embeddings = embed_texts(all_chunks, max_length=450)
```

```
# ────────────────────────────────────────────────────────────────────────────
# 2. DIMENSIONALITY REDUCTION WITH UMAP (CHUNK EMBEDDINGS)
# ────────────────────────────────────────────────────────────────────────────
import umap
import numpy as np

# Convert the PyTorch tensor to NumPy for UMAP
chunk_embeddings_np = chunk_embeddings.cpu().numpy()

# Configure UMAP to reduce our 450-token chunk embeddings to 5 dimensions
umap_reducer = umap.UMAP(
    n_neighbors=15,        # how many nearby chunks to consider
    n_components=5,        # target dimensionality
    metric="cosine",       # cosine distance on embedding space
    random_state=42
)

# Fit UMAP on chunk-level embeddings
umap_embeddings = umap_reducer.fit_transform(chunk_embeddings_np)
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/deprecation.py:151: FutureWarni
  warnings.warn(
/usr/local/lib/python3.11/dist-packages/umap/umap_.py:1952: UserWarning: n_jobs value
  warn(
```

```
# ────────────────────────────────────────────────────────────────────────────
# 3. CLUSTERING WITH HDBSCAN (ON UMAP-REDUCED CHUNKS)
# ────────────────────────────────────────────────────────────────────────────
import hdbscan

# HDBSCAN will discover dense topic clusters in the reduced UMAP space
clusterer = hdbscan.HDBSCAN(
    min_cluster_size=15,            # minimum chunks per cluster
    metric="euclidean",             # distance in UMAP space
    cluster_selection_method="eom" # cluster selection algorithm
)

# Fit & predict cluster labels for each chunk
cluster_labels = clusterer.fit_predict(umap_embeddings)
```

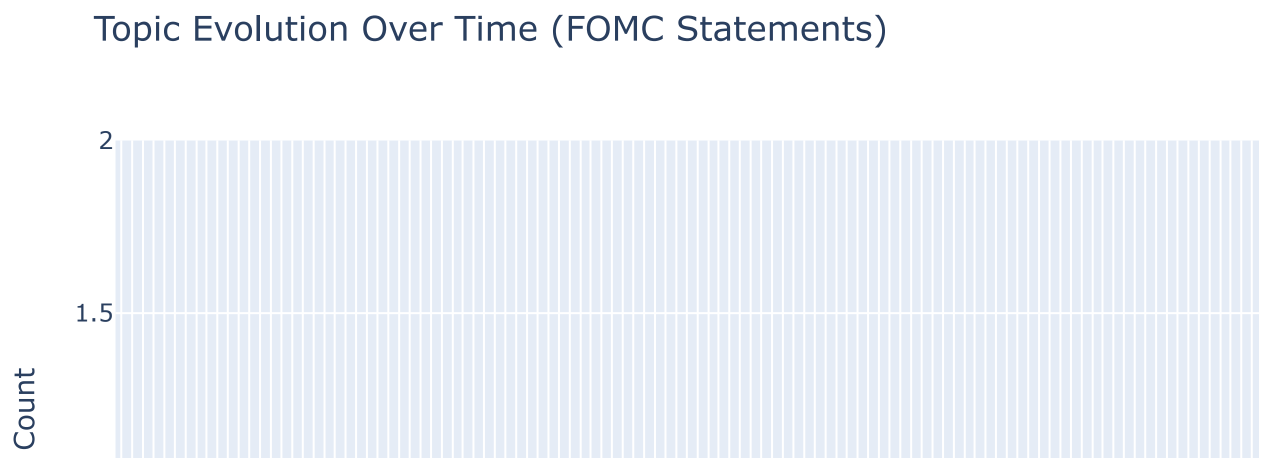
```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/deprecation.py:151: FutureWarni
  warnings.warn(
/usr/local/lib/python3.11/dist-packages/sklearn/utils/deprecation.py:151: FutureWarni
  warnings.warn(
```

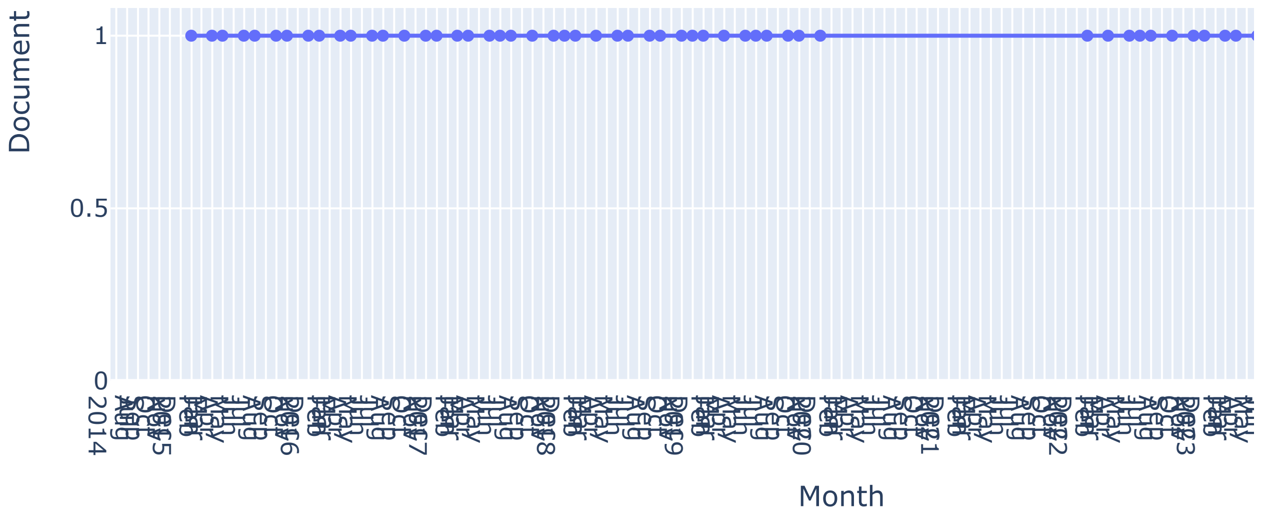
```
# ────────────────────────────────────────────────────────────────────────────
# 4. TOPIC REPRESENTATION USING MANUAL c-TF-IDF + MMR (NO BERTopic DEPENDENCY)
# ────────────────────────────────────────────────────────────────────────────
```

```
import numpy as np
from scipy.sparse import csr_matrix
from scipy.spatial.distance import cosine
from sklearn.feature_extraction.text import CountVectorizer

# STEP 1: Build Chunk-Term Matrix
vectorizer = CountVectorizer(stop_words="english")
dtm_chunks = vectorizer.fit_transform(all_chunks)  # (n_chunks × n_terms)
terms = vectorizer.get_feature_names_out().tolist()

# STEP 2: Manual c-TF-IDF Calculation
def compute_c_tf_idf(dtm: csr_matrix, labels: list[int]):
    """
    Compute class-based TF-IDF (c-TF-IDF) from a document-term matrix.
    """
    cluster_ids = sorted(set(labels))
    n_clusters = len(cluster_ids)

    tf = []
    for cid in cluster_ids:
        idx = [i for i, lbl in enumerate(labels) if lbl == cid]
        tf_c = np.array(dtm[idx, :].sum(axis=0)).reshape(-1)  # (n_terms,)
        tf.append(tf_c)
    tf = np.stack(tf)  # (n_clusters × n_terms)

    df = np.sum(tf > 0, axis=0)
    idf = np.log((n_clusters + 1) / (df + 1)) + 1

    return tf * idf, cluster_ids

ctfidf_matrix, cluster_list = compute_c_tf_idf(dtm_chunks, cluster_labels)

# STEP 3: Embed Terms (use existing FinBERT embed_texts from Block 1)
term_embeddings = embed_texts(terms, max_length=32).cpu().numpy()

# STEP 4: MMR Implementation
def mmr(candidates: list[str],
        scores: np.ndarray,
        embeddings: np.ndarray,
        top_n: int = 10,
        diversity: float = 0.7):
    """
    Select top_n terms balancing relevance and diversity using MMR.
    """
    selected, unselected = [], list(range(len(candidates)))

    min_s, max_s = scores.min(), scores.max()
    norm_scores = (scores - min_s) / (max_s - min_s + 1e-9)

    while len(selected) < min(top_n, len(unselected)):
        best_score, best_idx = None, None
        for idx in unselected:
            rel = norm_scores[idx]
            if selected:
                sim = max(1 - cosine(embeddings[idx], embeddings[s]) for s in selected)
            else:
                sim = 0
            mmr_score = diversity * rel - (1 - diversity) * sim
            if best_score is None or mmr_score > best_score:
                best_score, best_idx = mmr_score, idx
        selected.append(best_idx)
        unselected.remove(best_idx)
    return [candidates[i] for i in selected]

# STEP 5: Generate Topic Labels
topic_labels = {}
for i, cid in enumerate(cluster_list):
    if cid == -1:
        topic_labels[cid] = "Outliers"
        continue
    term_scores = ctfidf_matrix[i]
    top_terms = mmr(terms, term_scores, term_embeddings, top_n=10, diversity=0.7)
    topic_labels[cid] = " ".join(top_terms)



# ───────────────────────────────────────────────────────────────────────────
# 5. ASSIGN TOPICS TO CHUNKS & AGGREGATE BACK TO DOCUMENTS
# ───────────────────────────────────────────────────────────────────────────
import pandas as pd

# Build a DataFrame of chunks with their cluster labels
chunks_df = pd.DataFrame({
    "doc_idx":       [i for i, sub in enumerate(docs_chunks) for _ in sub],
    "chunk_text":    all_chunks,
    "cluster_id":    cluster_labels
})
# Map cluster ID → human-readable topic name
chunks_df["topic_label"] = chunks_df["cluster_id"].map(topic_labels)

# Aggregate chunk-level topics to each original document by majority vote
doc_topics = (
    chunks_df
    .groupby("doc_idx")["topic_label"]
    .agg(lambda lbls: lbls.value_counts().idxmax())
    .to_dict()
)
# Assign final topic per document in the original df
df["topics"] = df.index.to_series().map(doc_topics)
```

```
# ──────────────────────────────────────────────────────────────────────────────
# 6. PLOT TOPIC EVOLUTION OVER TIME WITH PLOTLY (USING 'Date' COLUMN)
# ──────────────────────────────────────────────────────────────────────────────
import plotly.express as px

# STEP 1: Parse the 'Date' column into datetime if not already
df["Date"] = pd.to_datetime(df["Date"])

# STEP 2: Group by Month × Topic
df_counts = (
    df.groupby([pd.Grouper(key="Date", freq="M"), "topics"])
      .size()
      .reset_index(name="count")
)

# STEP 3: Plot topic evolution using Plotly
fig = px.line(
    df_counts,
    x="Date",
    y="count",
    color="topics",
    markers=True,
    title="Topic Evolution Over Time (FOMC Statements)"
)

fig.update_layout(
    xaxis_title="Month",
    yaxis_title="Document Count",
    legend_title="Topics",
    xaxis=dict(dtick="M1", tickformat="%b\n%Y")  # Optional: month formatting
)

fig.show()
```

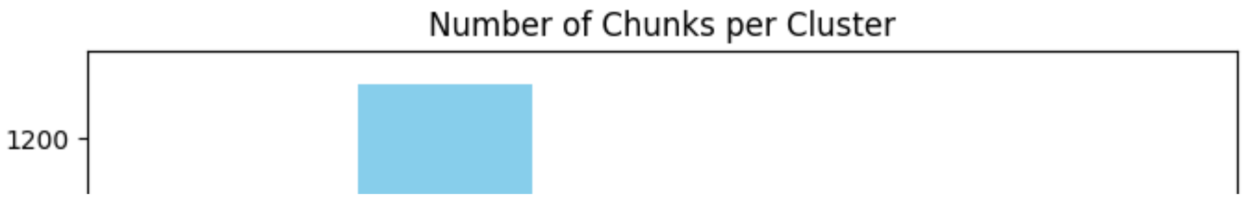
```
<ipython-input-9-862b2ab32b5f>:11: FutureWarning: 'M' is deprecated and will be remov
  df.groupby([pd.Grouper(key="Date", freq="M"), "topics"])
```

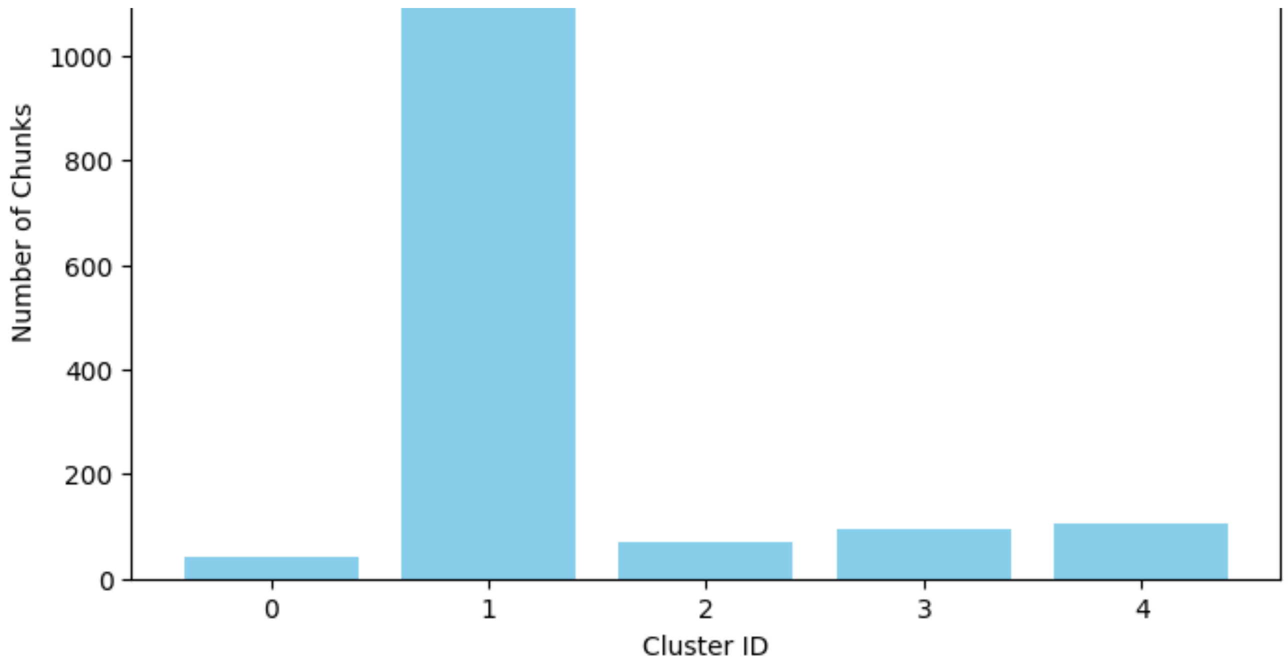




```
# ────────────────────────────────────────────────────────────────
# 6A. INSPECT CLUSTER DISTRIBUTION FROM HDBSCAN
# ────────────────────────────────────────────────────────────────
import numpy as np
import matplotlib.pyplot as plt
from collections import Counter

# Count how many chunks belong to each cluster label
cluster_summary = Counter(cluster_labels)

# Print label distribution
print("Cluster distribution:")
for label, count in cluster_summary.items():
    print(f"Cluster {label}: {count} chunks")

# Optional bar chart visualization
plt.figure(figsize=(8, 5))
plt.bar(cluster_summary.keys(), cluster_summary.values(), color="skyblue")
plt.xlabel("Cluster ID")
plt.ylabel("Number of Chunks")
plt.title("Number of Chunks per Cluster")
plt.show()
```

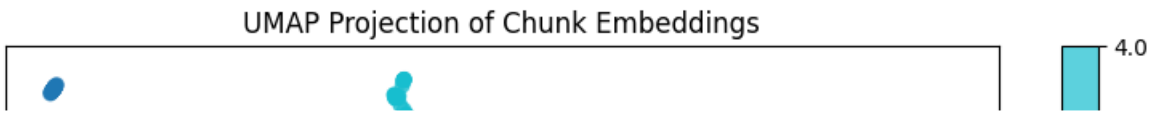
```
Cluster distribution:
Cluster 1: 1301 chunks
Cluster 4: 104 chunks
Cluster 3: 94 chunks
Cluster 2: 70 chunks
Cluster 0: 42 chunks
```

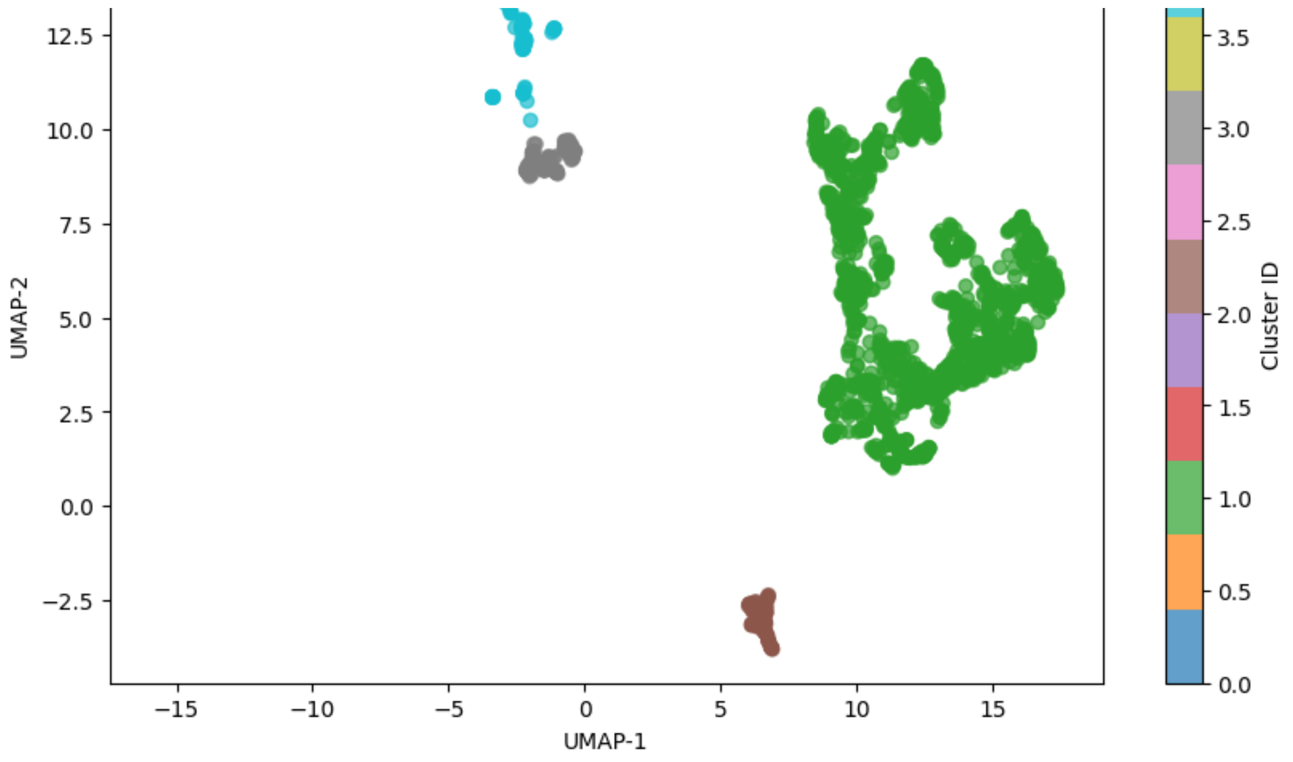




```
# ───────────────────────────────────────────────────────────────
# 6B. UMAP VISUALIZATION OF EMBEDDINGS (2D SCATTERPLOT)
# ───────────────────────────────────────────────────────────────
import umap

# Project chunk embeddings to 2D for visualization
umap_2d = umap.UMAP(n_components=2, metric="cosine", random_state=42)
points_2d = umap_2d.fit_transform(chunk_embeddings.cpu().numpy())

# Plot
plt.figure(figsize=(10, 6))
scatter = plt.scatter(points_2d[:, 0], points_2d[:, 1], c=cluster_labels, cmap='tab10', al
plt.colorbar(scatter, label="Cluster ID")
plt.title("UMAP Projection of Chunk Embeddings")
plt.xlabel("UMAP-1")
plt.ylabel("UMAP-2")
plt.show()
```

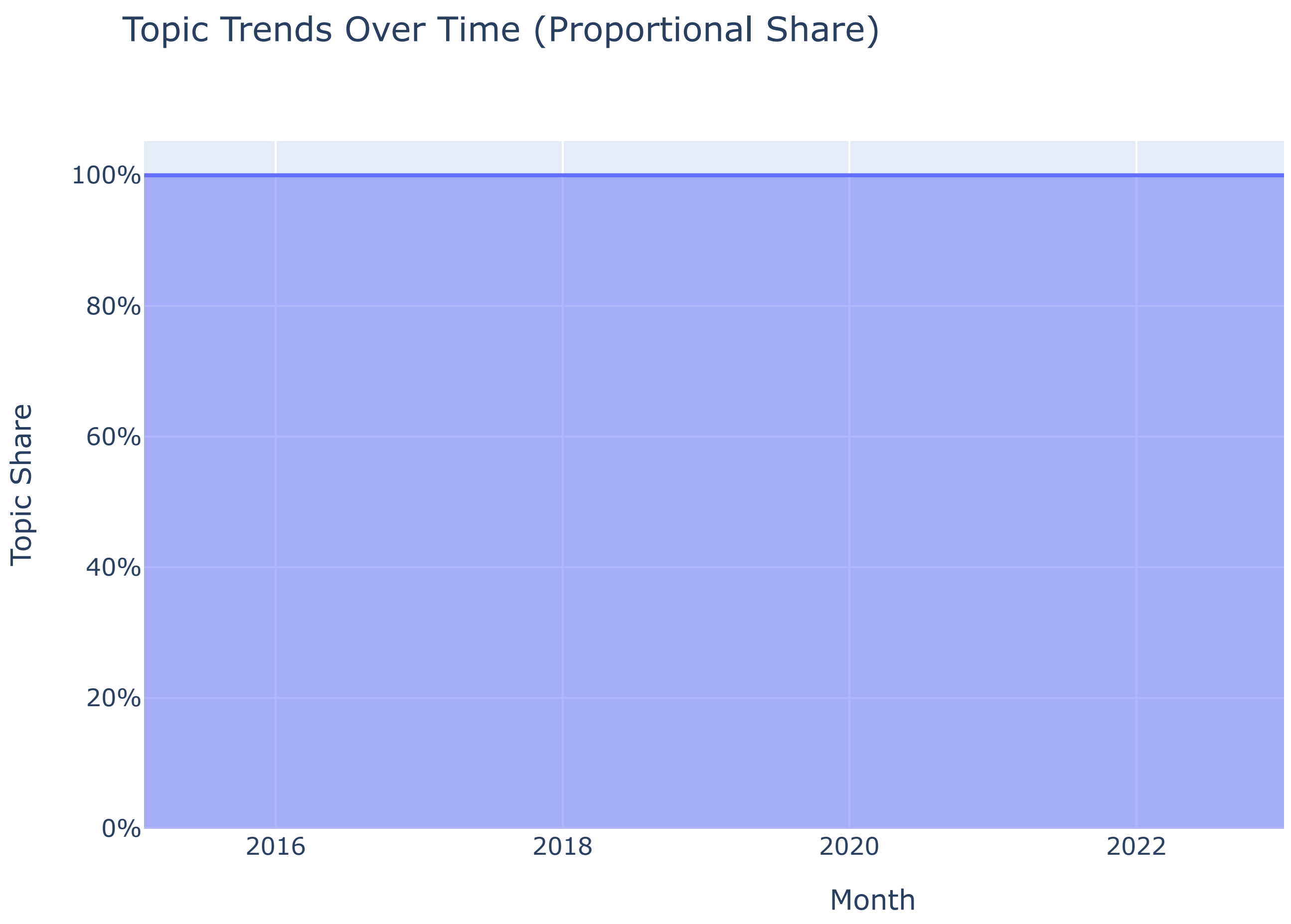
```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/deprecation.py:151: FutureWarni

'force_all_finite' was renamed to 'ensure_all_finite' in 1.6 and will be removed in 1

/usr/local/lib/python3.11/dist-packages/umap/umap_.py:1952: UserWarning:

n_jobs value 1 overridden to 1 by setting random_state. Use no seed for parallelism.
```





```
# ─────────────────────────────────────────────────────────────────────────────
# 7. STACKED AREA CHART: TOPIC TRENDS OVER TIME
# ─────────────────────────────────────────────────────────────────────────────
import plotly.express as px

# Make sure date column is datetime
df["Date"] = pd.to_datetime(df["Date"])

# Group by month and topic
df_counts = (
    df.groupby([pd.Grouper(key="Date", freq="M"), "topics"])
      .size()
      .reset_index(name="count")
)

# Normalize counts by month to get proportions
total_per_month = df_counts.groupby("Date")["count"].transform("sum")
df_counts["percentage"] = df_counts["count"] / total_per_month

# Stacked area plot to show relative topic dominance over time
fig = px.area(
    df_counts,
    x="Date",
    y="percentage",
    color="topics",
    title="Topic Trends Over Time (Proportional Share)",
    labels={"percentage": "Share of Topic"},
)

fig.update_layout(
    yaxis=dict(tickformat=".0%"),
    xaxis_title="Month",
    yaxis_title="Topic Share",
    legend_title="Topics"
)

fig.show()
```

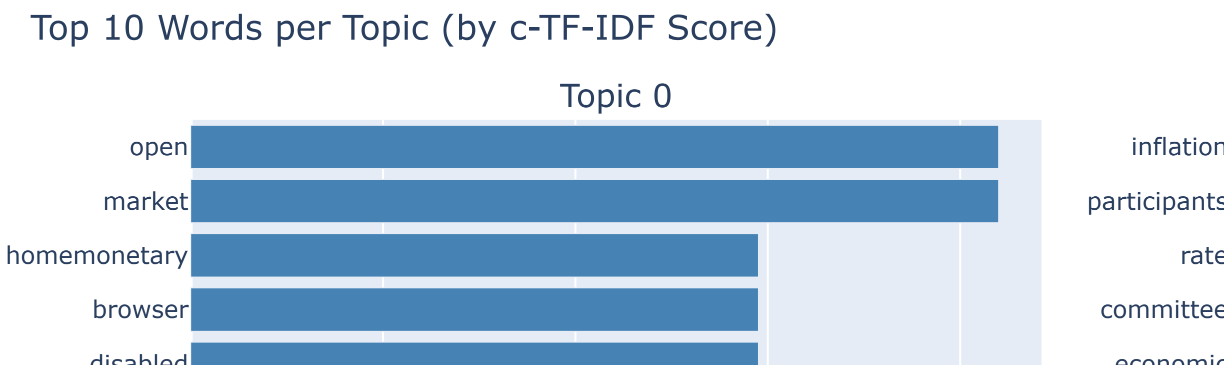
```
<ipython-input-12-86d82f90fec8>:11: FutureWarning:

'M' is deprecated and will be removed in a future version, please use 'ME' instead.
```

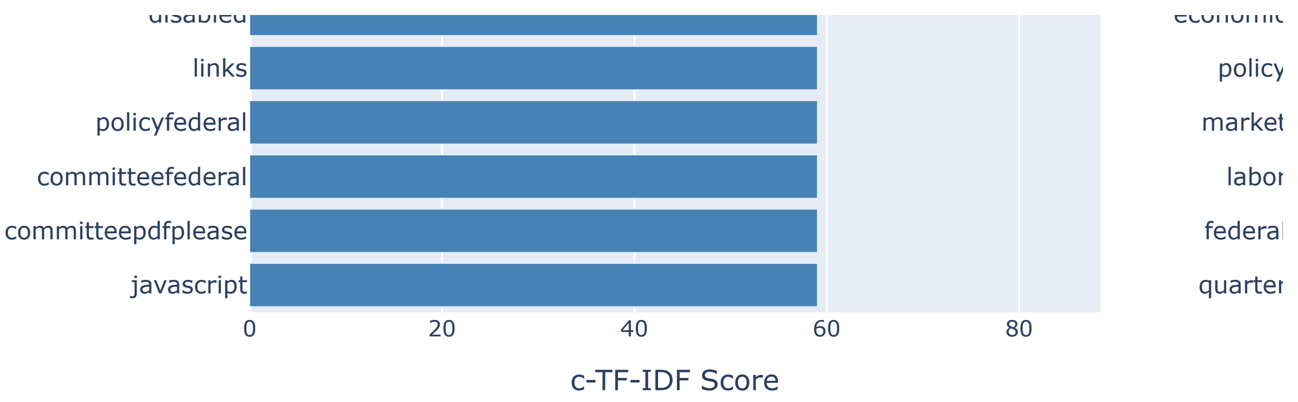


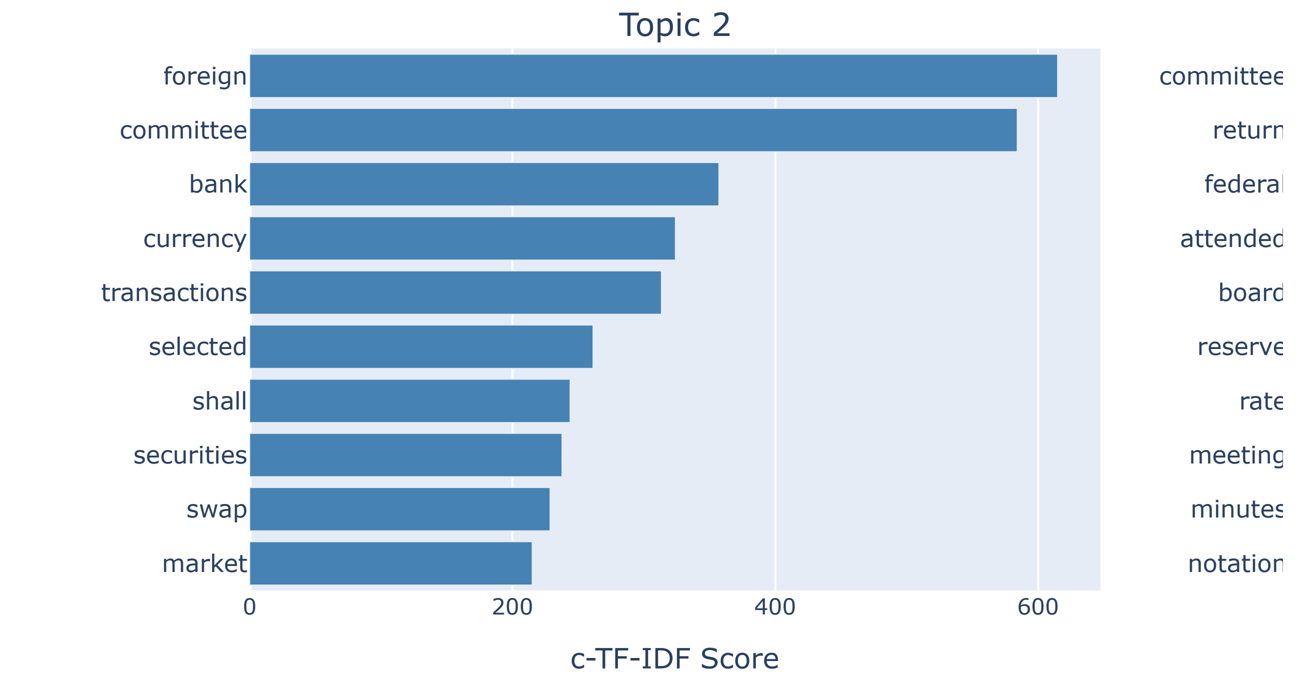
```
# ─────────────────────────────────────────────────────────────
# 8. SHOW TOP 10 TERMS PER TOPIC (FROM MMR OUTPUT) — DISPLAY + OPTIONAL EXPORT
# 8. SHOW TOP 10 TERMS PER TOPIC (FROM MMR OUTPUT) — DISPLAY + OPTIONAL EXPORT
# ────────────────────────────────────────────────────────────────────────────────
import pandas as pd

# Convert MMR results into a DataFrame
topic_words_df = pd.DataFrame([
    {"topic": topic, "top_words": words}
    for topic, words in topic_labels.items()
    if topic != "Outliers"
])

# Sort by topic ID (if numeric)
topic_words_df = topic_words_df.sort_values(by="topic").reset_index(drop=True)

# Print table
print("Top 10 Terms per Topic:")
print(topic_words_df.to_string(index=False))

# Optional: Save to CSV
# topic_words_df.to_csv("topic_keywords.csv", index=False)
```

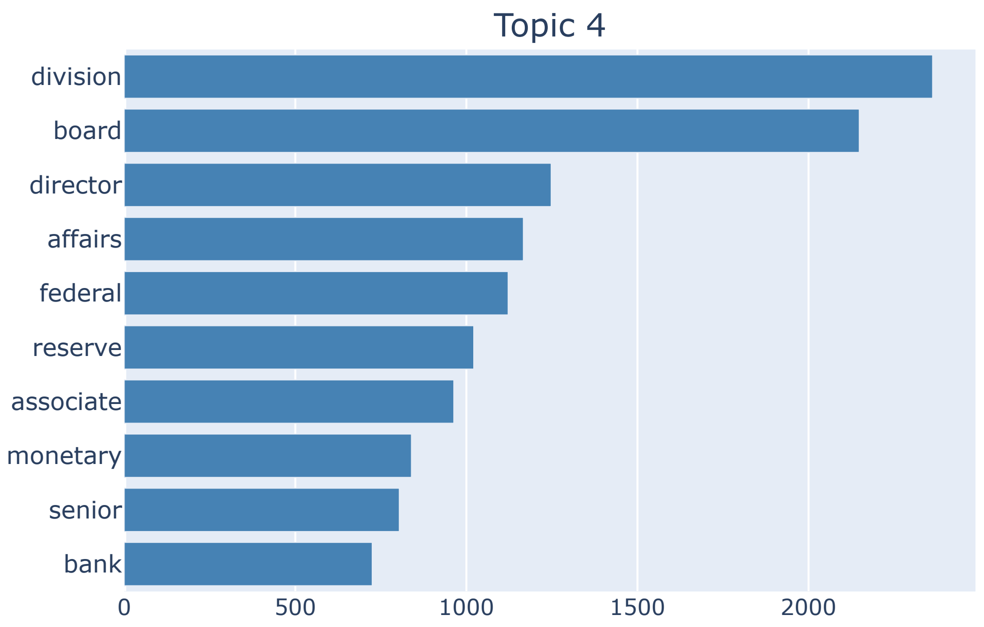
```
Top 10 Terms per Topic:
 topic
     0 market open committeepdfplease committeefederal javascript disabled homemoneta
     1                           inflation participants rate economic market percent
     2                           foreign committee bank transactions currency selecte
     3                                committee return federal attended board reserve
     4                               division board federal affairs director reserve
```

```
# ────────────────────────────────────────────────────────────────────────────────
# 9. PLOTLY SUBPLOTS (RESIZED): TOP 10 TERMS FOR EACH TOPIC
# ────────────────────────────────────────────────────────────────────────────────
import plotly.graph_objects as go
from plotly.subplots import make_subplots

top_n = 10
topics_to_plot = [cid for cid in cluster_list if cid != -1]
topics_to_plot = sorted(set(topics_to_plot))
num_topics = len(topics_to_plot)

# Layout parameters
cols = 2
rows = (num_topics + cols - 1) // cols
subplot_height = 400  # increased height per row
subplot_width = 1100  # increased width for better label spacing

fig = make_subplots(
    rows=rows, cols=cols,
    subplot_titles=[f"Topic {cid}" for cid in topics_to_plot],
    vertical_spacing=0.12
)

# Plot each topic
for i, topic_id in enumerate(topics_to_plot):
    row = (i // cols) + 1
    col = (i % cols) + 1

    term_scores = ctfidf_matrix[cluster_list.index(topic_id)]
    top_indices = term_scores.argsort()[::-1][:top_n]
    words = [terms[i] for i in top_indices]
    scores = [term_scores[i] for i in top_indices]

    fig.add_trace(
        go.Bar(
            x=scores[::-1],
            y=words[::-1],
            orientation="h",
            name=f"Topic {topic_id}",
            showlegend=False,
            marker_color='steelblue'
        ),
        row=row, col=col
    )

# Final layout adjustments
fig.update_layout(
    height=rows * subplot_height,
    width=subplot_width,
    title_text="Top 10 Words per Topic (by c-TF-IDF Score)",
    margin=dict(t=80, l=60, r=20, b=40),
    font=dict(size=12)
)

fig.update_yaxes(tickfont=dict(size=12))
fig.update_xaxes(title_text="c-TF-IDF Score", tickfont=dict(size=11))

fig.show()
```




disabled
links
policyfederal
committeefederal
committeepdfplease
javascript
0
20
40
60
80
c-TF-IDF Score
economic
policy
market
labor
federal
quarter


Topic 2
foreign
committee
bank
currency
transactions
selected
shall
securities
swap
market
0
200
400
600
c-TF-IDF Score
committee
return
federal
attended
board
reserve
rate
meeting
minutes
notation


Topic 4
division
board
director
affairs
federal
reserve
associate
monetary
senior
bank
0
500
1000
1500
2000

c-TF-IDF Score

```
df.sample(2)
```

	URL	Date	Year	Month	Day	Content	date
1	https://www.federalreserve.gov/monetarypolicy/...	2025-01-29	2025	2025-01-01	29	HomeMonetary PolicyFederal Open Market Committ...	2025-01-31
46	https://www.federalreserve.gov/monetarypolicy/...	2017-06-14	2017	2017-06-01	14	HomeMonetary PolicyFederal Open Market Committ...	2017-06-30

2 rows × 21 columns

```
# Assuming df is your DataFrame
df.to_csv('/content/drive/My Drive/NLP/Assignment_3/fomc_topic.csv', index=False)
```

Multi-topic assignment per Topic

```
# ──────────────────────────────────────────────────────────────────────────────
# MULTI-TOPIC ASSIGNMENT PER DOCUMENT (TOP 3 BY CHUNK FREQUENCY)
# ──────────────────────────────────────────────────────────────────────────────
import pandas as pd

# Step 1: Rebuild the chunks DataFrame if needed
chunks_df = pd.DataFrame({
    "doc_idx": [i for i, sub in enumerate(docs_chunks) for _ in sub],
    "chunk_text": all_chunks,
    "cluster_id": cluster_labels
})
chunks_df["topic_label"] = chunks_df["cluster_id"].map(topic_labels)

# Step 2: Group chunks by document and assign top N topics by frequency
top_n = 3
multi_topic_map = (
    chunks_df.groupby("doc_idx")["topic_label"]
    .agg(lambda x: x.value_counts().index[:top_n].tolist())
    .to_dict()
    .to_dict()
)

# Step 3: Add multi-topic list to original DataFrame
df["multi_topics"] = df.index.to_series().map(lambda i: multi_topic_map.get(i, []))

# Step 4: Explode to one topic per row (for plotting)
df_exploded = df.explode("multi_topics").rename(columns={"multi_topics": "topics"})

# Step 5 (Optional): View how many docs have multiple topics
df["num_topics_assigned"] = df["multi_topics"].apply(len)
print(df["num_topics_assigned"].value_counts())
```

```
num_topics_assigned
3    66
Name: count, dtype: int64
```

```
df.sample(2)
```

	URL	Date	Year	Month	Day	Content	date
0	https:// www.federalreserve.gov/ monetarypolicy/...	2025-03-19	2025	2025-03-01	19	HomeMonetary PolicyFederal Open Market Committ...	2025-03-31
32	https:// www.federalreserve.gov/ monetarypolicy/...	2019-03-20	2019	2019-03-01	20	HomeMonetary PolicyFederal Open Market Committ...	2019-03-31

2 rows × 23 columns