Module 5 Assignment: Topic Modeling on COVID-19 Research Papers

This assignment focuses on building and interpreting topic models using a dataset of research papers related to COVID-19, published before the 2020 pandemic. we will preprocess the text, vectorize it, build topic models, evaluate them, and interpret the topics.

Assignment Steps:

1. Pre-process the data (remove stop words, apply stemming/lemmatization)
2. Vectorize the text using `CountVectorizer` (optionally with bigrams/trigrams)
3. Build topic models with 8, 9, and 10 topics
4. Evaluate models using perplexity and/or topic coherence
5. Select and interpret a model, describe topics, and visualize results
6. Summarize your findings

```
import nltk
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /Users/balaji/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/balaji/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True
```

```
# Reinstall numpy, scipy, and gensim to fix binary incompatibility
!pip install --force-reinstall --no-cache-dir numpy scipy gensim
```

```
Downloading numpy-1.26.4-cp311-cp311-macosx_11_0_arm64.whl (14.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 14.0/14.0 MB 11.8 MB/s eta 0:00:00a 0:00:01
Downloading scipy-1.13.1-cp311-cp311-macosx_12_0_arm64.whl (30.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 14.0/14.0 MB 11.8 MB/s eta 0:00:00
Downloading scipy-1.13.1-cp311-cp311-macosx_12_0_arm64.whl (30.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 30.3/30.3 MB 10.8 MB/s eta 0:00:00a 0:00:01
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 30.3/30.3 MB 10.8 MB/s eta 0:00:00
Downloading smart_open-7.1.0-py3-none-any.whl (61 kB)
Downloading wrapt-1.17.2-cp311-cp311-macosx_11_0_arm64.whl (38 kB)
Downloading smart_open-7.1.0-py3-none-any.whl (61 kB)
Downloading wrapt-1.17.2-cp311-cp311-macosx_11_0_arm64.whl (38 kB)
Installing collected packages: wrapt, numpy, smart-open, scipy, gensim
  Attempting uninstall: wrapt
    Found existing installation: wrapt 1.17.2
    Uninstalling wrapt-1.17.2:
      Successfully uninstalled wrapt-1.17.2
  Attempting uninstall: numpy
    Found existing installation: numpy 1.26.4
Installing collected packages: wrapt, numpy, smart-open, scipy, gensim
  Attempting uninstall: wrapt
    Found existing installation: wrapt 1.17.2
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5/5 [gensim]2m4/5 [gensim]
   Successfully installed gensim-4.3.3 numpy-1.26.4 scipy-1.13.1 smart-open-7.1.0 wrapt-1.17.2
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5/5 [gensim]
   Successfully installed gensim-4.3.3 numpy-1.26.4 scipy-1.13.1 smart-open-7.1.0 wrapt-1.17.2
```

```
# Remove all __pycache__ folders and force reinstall numpy and gensim
import os
import shutil

for root, dirs, files in os.walk(".", topdown=False):
    for name in dirs:
        if name == "__pycache__":
            shutil.rmtree(os.path.join(root, name))

# Now force reinstall numpy and gensim
!pip install --force-reinstall --no-cache-dir numpy gensim
```

```
Collecting numpy
  Downloading numpy-2.3.0-cp311-cp311-macosx_14_0_arm64.whl.metadata (62 kB)
  Downloading numpy-2.3.0-cp311-cp311-macosx_14_0_arm64.whl.metadata (62 kB)
Collecting gensim
  Downloading gensim-4.3.3-cp311-cp311-macosx_11_0_arm64.whl.metadata (8.1 kB)
Collecting numpy
Collecting gensim
  Downloading gensim-4.3.3-cp311-cp311-macosx_11_0_arm64.whl.metadata (8.1 kB)
Collecting numpy
  Downloading numpy-1.26.4-cp311-cp311-macosx_11_0_arm64.whl.metadata (114 kB)
  Downloading numpy-1.26.4-cp311-cp311-macosx_11_0_arm64.whl.metadata (114 kB)
Collecting scipy<1.14.0,>=1.7.0 (from gensim)
  Downloading scipy-1.13.1-cp311-cp311-macosx_12_0_arm64.whl.metadata (60 kB)
Collecting scipy<1.14.0,>=1.7.0 (from gensim)
  Downloading scipy-1.13.1-cp311-cp311-macosx_12_0_arm64.whl.metadata (60 kB)
Collecting smart-open>=1.8.1 (from gensim)
  Downloading smart_open-7.1.0-py3-none-any.whl.metadata (24 kB)
Collecting smart-open>=1.8.1 (from gensim)
  Downloading smart_open-7.1.0-py3-none-any.whl.metadata (24 kB)
Collecting wrapt (from smart-open>=1.8.1->gensim)
  Downloading wrapt-1.17.2-cp311-cp311-macosx_11_0_arm64.whl.metadata (6.4 kB)
Downloading gensim-4.3.3-cp311-cp311-macosx_11_0_arm64.whl (24.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0.0/24.0 MB ? eta -:--:--Collecting wrapt (from smart-open>=1.8.1->gensim)
  Downloading wrapt-1.17.2-cp311-cp311-macosx_11_0_arm64.whl.metadata (6.4 kB)
Downloading gensim-4.3.3-cp311-cp311-macosx_11_0_arm64.whl (24.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.0/24.0 MB 11.5 MB/s eta 0:00:0000:0100:01
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.0/24.0 MB 11.5 MB/s eta 0:00:0000:01
Downloading numpy-1.26.4-cp311-cp311-macosx_11_0_arm64.whl (14.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0.0/14.0 MB ? eta -:--:--Downloading numpy-1.26.4-cp311-cp311-macosx_11_0_ar
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 14.0/14.0 MB 13.7 MB/s eta 0:00:00a 0:00:01
Downloading scipy-1.13.1-cp311-cp311-macosx_12_0_arm64.whl (30.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 14.0/14.0 MB 13.7 MB/s eta 0:00:00
Downloading scipy-1.13.1-cp311-cp311-macosx_12_0_arm64.whl (30.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 30.3/30.3 MB 12.4 MB/s eta 0:00:00a 0:00:01
Downloading smart_open-7.1.0-py3-none-any.whl (61 kB)
Downloading wrapt-1.17.2-cp311-cp311-macosx_11_0_arm64.whl (38 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 30.3/30.3 MB 12.4 MB/s eta 0:00:00
Downloading smart_open-7.1.0-py3-none-any.whl (61 kB)
Downloading wrapt-1.17.2-cp311-cp311-macosx_11_0_arm64.whl (38 kB)
Installing collected packages: wrapt, numpy, smart-open, scipy, gensim
  Attempting uninstall: wrapt
    Found existing installation: wrapt 1.17.2
    Uninstalling wrapt-1.17.2:
      Successfully uninstalled wrapt-1.17.2
  Attempting uninstall: numpy
    Found existing installation: numpy 1.26.4
Installing collected packages: wrapt, numpy, smart-open, scipy, gensim
  Attempting uninstall: wrapt
    Found existing installation: wrapt 1.17.2
    Uninstalling wrapt-1.17.2:
      Successfully uninstalled wrapt-1.17.2
  Attempting uninstall: numpy
    Found existing installation: numpy 1.26.4
    Uninstalling numpy-1.26.4:
      Successfully uninstalled numpy-1.26.4
    Uninstalling numpy-1.26.4:
      Successfully uninstalled numpy-1.26.4
  Attempting uninstall: smart-openm━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1/5 [numpy]
```

```
# Import required libraries
import os
import glob
import json
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from helper_functions import *


# Load the dataset (using a subset for demonstration if needed)
data_dir = "./module-5/dataset/biorxiv_medrxiv/biorxiv_medrxiv/"
file_list = glob.glob(os.path.join(data_dir, '*.json'))

# Read a subset of files for faster processing (adjust n_files as needed)
n_files = 100  # You can increase this if your machine can handle more
papers = []
for file in file_list[:n_files]:
    with open(file, 'r') as f:
        paper = json.load(f)
        papers.append(paper)

# Extract text (e.g., title + abstract)
def extract_text(paper):
    title = paper.get('title', '')
    abstract = paper.get('abstract', '')
    if isinstance(abstract, list):
        abstract = ' '.join([a.get('text', '') for a in abstract])
    return f"{title} {abstract}"

documents = [extract_text(p) for p in papers]

# Preview a few documents
for i, doc in enumerate(documents[:3]):
    print(f"Document {i+1}:\n", doc[:500], "\n---\n")
```

Data Loading: All Subfolders

We will load data from all four JSON subfolders using the provided `load_data` helper function. This ensures our topic model covers the full dataset.

```
# Load data from all four subfolders using the helper function
# This will return a list of documents (strings)

from helper_functions import load_data
import glob
import os

# Use absolute paths relative to the notebook location
base_path = os.path.abspath('dataset/')
data_folders = [
    os.path.join(base_path, 'biorxiv_medrxiv/biorxiv_medrxiv/'),
    os.path.join(base_path, 'comm_use_subset/comm_use_subset/'),
    os.path.join(base_path, 'custom_license/custom_license/'),
    os.path.join(base_path, 'noncomm_use_subset/noncomm_use_subset/')
]

all_documents = []
for folder in data_folders:
    files = glob.glob(os.path.join(folder, '*.json'))
    print(f"Absolute path: {folder}")
    print(f"Found {len(files)} JSON files in {folder}")
    if len(files) > 0:
        docs = load_data(folder, n_files=5)  # Try with 5 for quick test
        print(f"Loaded {len(docs)} documents from {folder}")
        all_documents.extend(docs)
    else:
        print(f"No JSON files found in {folder}. Please check the path and data.")

print(f"Total loaded documents: {len(all_documents)}")
if all_documents:
    print("Sample document:\n", all_documents[0][:500])
else:
    print("No documents loaded. Please check your folder paths and data.")
```

```
Absolute path: /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/module-5/datase
Found 885 JSON files in /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/module
```

```
Loaded 5 documents from /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/module
Absolute path: /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/module-5/datase
Found 9118 JSON files in /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/modul
Loaded 5 documents from /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/module
Absolute path: /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/module-5/datase
Found 16959 JSON files in /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/modu
Loaded 5 documents from /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/module
Absolute path: /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/module-5/datase
Found 2353 JSON files in /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/modul
Loaded 5 documents from /Users/balaji/source/san-diego/assignments/Machine-learning-Fundamentals-and-Applications/module
Total loaded documents: 20
Sample document:
 Multimerization of HIV-1 integrase hinges on conserved SH3-docking platforms New anti-AIDS treatments must be continual
```

```
# Test helper functions to ensure they work
# We'll use a small sample for demonstration
sample_docs = all_documents[:5]

# Display topics (dummy LDA for test)
vectorizer = CountVectorizer(stop_words='english', max_df=0.95, min_df=2)
X = vectorizer.fit_transform(sample_docs)
lda = LatentDirichletAllocation(n_components=2, random_state=42)
lda.fit(X)

# Import display_topics from helper_functions and use it
from helper_functions import display_topics
display_topics(lda, vectorizer.get_feature_names_out(), 5)
```

```
Topic 1:
  cov sars samples 2020 viral

Topic 2:
  cases figure site et al
```

Text Preprocessing: Stop Words Removal and Lemmatization

We will preprocess the documents by removing stop words and applying lemmatization. This step helps to normalize the text and improve topic modeling quality.

```
from nltk.corpus import stopwords
import re

# Prepare stop words
stop_words = set(stopwords.words('english'))

# Use stemming instead of lemmatization to avoid wordnet dependency
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()

def preprocess_text(text):
    # Tokenize using regex (splits on word boundaries)
    tokens = re.findall(r'\b\w+\b', text.lower())
    # Remove stop words and stem
    tokens = [stemmer.stem(word) for word in tokens if word.isalpha() and word not in stop_words]
    return ' '.join(tokens)

# Apply preprocessing to all documents
preprocessed_documents = [preprocess_text(doc) for doc in all_documents]

# Preview a preprocessed document
print(preprocessed_documents[0][:500])
```

```
multimer hiv integras hing conserv dock platform new anti aid treatment must continu develop order overcom resist mutat
```

Vectorization: CountVectorizer with Bigrams/Trigrams Option

We will vectorize the preprocessed text using CountVectorizer. You can adjust ngram_range for unigrams, bigrams, or trigrams.

```
from sklearn.feature_extraction.text import CountVectorizer

# You can adjust ngram_range to (1,2) for bigrams or (1,3) for trigrams
vectorizer = CountVectorizer(max_df=0.95, min_df=2, ngram_range=(1,2))
X = vectorizer.fit_transform(preprocessed_documents)

print(f"Document-term matrix shape: {X.shape}")
```

```
Document-term matrix shape: (20, 3538)
```

Topic Modeling: Train LDA Models for 8, 9, and 10 Topics

We will train LDA models for different topic counts and evaluate them using perplexity.

```
from sklearn.decomposition import LatentDirichletAllocation

n_topics_list = [8, 9, 10]
lda_models = {}
perplexities = {}

for n_topics in n_topics_list:
    lda = LatentDirichletAllocation(n_components=n_topics, random_state=42, learning_method='batch')
    lda.fit(X)
    lda_models[n_topics] = lda
    perplexity = lda.perplexity(X)
    perplexities[n_topics] = perplexity
    print(f"LDA with {n_topics} topics: Perplexity = {perplexity:.2f}")
```

```
LDA with 8 topics: Perplexity = 1160.01
LDA with 9 topics: Perplexity = 1137.09
LDA with 10 topics: Perplexity = 1151.82
LDA with 10 topics: Perplexity = 1151.82
```

Topic Model Evaluation and Selection

We compare perplexity scores to select the best topic count. Lower perplexity indicates a better model fit.

```
# Display perplexity scores for each model
for n_topics, perp in perplexities.items():
    print(f"Topics: {n_topics}, Perplexity: {perp:.2f}")

# Select the best model (lowest perplexity)
best_n_topics = min(perplexities, key=perplexities.get)
best_lda = lda_models[best_n_topics]
print(f"\nSelected {best_n_topics} topics as the best model.")
```

```
Topics: 8, Perplexity: 1160.01
Topics: 9, Perplexity: 1137.09
Topics: 10, Perplexity: 1151.82

Selected 9 topics as the best model.
```

Topic Interpretation: Top Words and Descriptions

We extract the top words for each topic and provide a brief interpretation for each.

```
from helper_functions import display_topics

# Display top words for each topic in the best model
display_topics(best_lda, vectorizer.get_feature_names_out(), 10)
```

```
Topic 1:
  al et al et cell virus viral estim activ sampl use

Topic 2:
  cov season sar sampl sar cov patient viru viral peak transmiss

Topic 3:
  hong hong kong kong polic particip plasma day studi care influenza

Topic 4:
  intern co per per cent cent intens growth increas global demand

Topic 5:
  protein express gene immun optim codon vaccin challeng protect day

Topic 6:
  cell protein viral dna rna mrna ribosom viru virus translat

Topic 7:
  case preprint doi author number estim time peer peer review review

Topic 8:
  cell protein neutral viru antibodi infect use ml express incub
```

```
Topic 9:
  cell ifn viru infect candid strain murin human mice ml
```

Topic Descriptions

Write 1–2 sentences describing the main theme of each topic based on the top words above. What makes each topic stand out?

1. **Topic 1:** This topic focuses on general virology and laboratory methods, with frequent mentions of 'cell', 'virus', 'viral', and 'sample'. It likely covers experimental studies and viral activity measurements.
2. **Topic 2:** This topic is centered on coronaviruses, especially SARS and seasonal coronaviruses, with terms like 'cov', 'sar', 'patient', and 'transmiss'. It likely discusses patient data and transmission dynamics.
3. **Topic 3:** This topic appears to relate to studies in Hong Kong, with references to 'hong kong', 'polic', 'particip', and 'care', possibly covering epidemiological studies or public health responses in that region.
4. **Topic 4:** This topic is about international and global trends, with words like 'intern', 'per cent', 'growth', and 'global demand', suggesting a focus on global health, resource needs, or epidemiological trends.
5. **Topic 5:** This topic is focused on immunology and vaccine development, with terms like 'protein', 'express', 'gene', 'vaccin', and 'protect', indicating studies on immune response and vaccine efficacy.
6. **Topic 6:** This topic covers molecular biology, especially genetic material and translation, with words like 'cell', 'protein', 'dna', 'rna', 'mrna', and 'translat', likely discussing viral replication and gene expression.
7. **Topic 7:** This topic is about publication and peer review, with terms like 'case', 'preprint', 'doi', 'author', 'peer review', and 'time', indicating meta-scientific or bibliometric analysis.
8. **Topic 8:** This topic focuses on neutralization and antibody studies, with words like 'cell', 'protein', 'neutral', 'antibodi', 'infect', and 'incub', likely covering laboratory assays and immune response.
9. **Topic 9:** This topic is about animal models and infection studies, with terms like 'cell', 'ifn', 'infect', 'strain', 'murin', 'mice', and 'human', suggesting research on infection mechanisms in animal models.

Overall Summary

Summarize all topics in 1–2 sentences, highlighting the diversity or focus of the research papers.