```
pip install nltk spacy
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.12/dist-packages (3.9.1)
Requirement already satisfied: spacy in /usr/local/lib/python3.12/dist-packages (3.8.11)
Requirement already satisfied: click in /usr/local/lib/python3.12/dist-packages (from nltk) (8.3.
Requirement already satisfied: joblib in /usr/local/lib/python3.12/dist-packages (from nltk) (1.5
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.12/dist-packages (from n
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (from nltk) (4.67.
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.12/dist-pack
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.12/dist-pack
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.12/dist-packages (fr
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.12/dist-packages (
Requirement already satisfied: thinc<8.4.0,>=8.3.4 in /usr/local/lib/python3.12/dist-packages (fr
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.12/dist-packages (f
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.12/dist-packages (fr
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: weasel<0.5.0,>=0.4.2 in /usr/local/lib/python3.12/dist-packages (f
Requirement already satisfied: typer-slim<1.0.0,>=0.3.0 in /usr/local/lib/python3.12/dist-package
Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.12/dist-packages (from spa
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.12/
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from spacy) (3.
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from spacy)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from s
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: pydantic-core==2.41.4 in /usr/local/lib/python3.12/dist-packages (
Requirement already satisfied: typing-extensions>=4.14.1 in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: typing-inspection>=0.4.2 in /usr/local/lib/python3.12/dist-package
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-package
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requ
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (fro
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (fro
Requirement already satisfied: blis<1.4.0,>=1.3.0 in /usr/local/lib/python3.12/dist-packages (fro
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.12/dist-package
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/python3.12/dist-package
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from j
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from smart-open<
```

```
import nltk
import spacy

print("NLTK and spaCy have been successfully imported.")
```

```
NLTK and spaCy have been successfully imported.
```

Natural Language Processing (NLP) is a field of artificial intelligence that enables computers to understand, interpret, and generate human language. It combines computational linguistics—rule-based modeling of human language—with statistical, machine learning, and deep learning models. Together, these technologies enable computers to process human language in the form of text or voice data and to 'understand' its full meaning, including the speaker's or writer's intent and sentiment.

```
essay_text = """
Natural Language Processing (NLP) is a field of artificial intelligence that enables computers t
"""

print(essay_text[:100] + "...") # Print first 100 characters to verify
```

```
Natural Language Processing (NLP) is a field of artificial intelligence that enables computers to
```

```
from nltk.tokenize import word_tokenize
import nltk

# Download the 'punkt' and 'punkt_tab' tokenizers if not already downloaded
nltk.download('punkt')
nltk.download('punkt_tab')

words_nltk = word_tokenize(essay_text)

print(f"Total words (including punctuation): {len(words_nltk)}")
print("First 20 words:")
print(words_nltk[:20])
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
Total words (including punctuation): 82
First 20 words:
['Natural', 'Language', 'Processing', '(', 'NLP', ')', 'is', 'a', 'field', 'of', 'artificial', 'i
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
```

```
import nltk

# Download the 'averaged_perceptron_tagger_eng' if not already downloaded
nltk.download('averaged_perceptron_tagger_eng')

# Apply POS tagging
pos_tags_nltk = nltk.pos_tag(words_nltk)

print("First 20 POS tagged words:")
print(pos_tags_nltk[:20])

# Get the unique set of POS tags
unique_tags = sorted(list(set([tag for word, tag in pos_tags_nltk])))
print(f"\nUnique NLTK POS Tags ({len(unique_tags)} tags):\n{unique_tags}")
```

```
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.
First 20 POS tagged words:
[('Natural', 'JJ'), ('Language', 'NNP'), ('Processing', 'NNP'), ('(', '('), ('NLP', 'NNP'), (')',

Unique NLTK POS Tags (21 tags):
["''", '(', ')', ',', '.', 'CC', 'DT', 'IN', 'JJ', 'NN', 'NNP', 'NNS', 'PRP', 'PRP$', 'RB', 'TO',
```

```
!python -m spacy download en_core_web_sm
```

```
Collecting en-core-web-sm==3.8.0
  Downloading https://github.com/explosion/spacy-models/releases/download/en_core_web_sm-3.8.0/en
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.8/12.8 MB 74.8 MB/s eta 0:00:00
```

```
✓ Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
⚠ Restart to reload dependencies
If you are in a Jupyter or Colab notebook, you may need to restart Python in
order to load all the package's dependencies. You can do this by selecting the
'Restart kernel' or 'Restart runtime' option.
```

```
print("\n--- Comparison of NLTK and spaCy POS Tags ---\n")

print("NLTK Unique Tags (fine-grained):")
print(unique_tags)
print(f"Number of NLTK tags: {len(unique_tags)}\n")

print("spaCy Unique Universal Tags:")
print(unique_universal_tags)
print(f"Number of spaCy universal tags: {len(unique_universal_tags)}\n")

# Example of mapping NLTK to Universal Tags (simplified)
print("Observations:")
print("- NLTK provides more granular tags (e.g., NNP, NNS, NN for nouns; VB, VBP, VBZ for verbs)
print("- spaCy's universal tags are broader categories (e.g., all nouns fall under NOUN, all ver
print("- Punctuation and spaces are also tagged differently (NLTK uses symbols like '(', ',', '.
```

```
--- Comparison of NLTK and spaCy POS Tags ---

NLTK Unique Tags (fine-grained):
["''", '(', ')', ',', '.', 'CC', 'DT', 'IN', 'JJ', 'NN', 'NNP', 'NNS', 'PRP', 'PRP$', 'RB', 'TO',
Number of NLTK tags: 21

spaCy Unique Universal Tags:
['ADJ', 'ADP', 'ADV', 'AUX', 'CCONJ', 'DET', 'NOUN', 'PART', 'PRON', 'PROPN', 'PUNCT', 'SPACE', '
Number of spaCy universal tags: 13

Observations:
- NLTK provides more granular tags (e.g., NNP, NNS, NN for nouns; VB, VBP, VBZ for verbs).
- spaCy's universal tags are broader categories (e.g., all nouns fall under NOUN, all verbs under
- Punctuation and spaces are also tagged differently (NLTK uses symbols like '(', ',', '.', etc.;
```

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

# Extract tokens and their universal POS tags
pos_tags_spacy = [(token.text, token.pos_) for token in doc]

print("First 20 POS tagged words (spaCy, Universal Tags):")
print(pos_tags_spacy[:20])

# Get the unique set of universal POS tags
unique_universal_tags = sorted(list(set([tag for word, tag in pos_tags_spacy])))
print(f"\nUnique spaCy Universal POS Tags ({len(unique_universal_tags)} tags):\n{unique_universa
```

```
First 20 POS tagged words (spaCy, Universal Tags):
[('\n', 'SPACE'), ('Natural', 'PROPN'), ('Language', 'PROPN'), ('Processing', 'PROPN'), ('(', 'PU

Unique spaCy Universal POS Tags (13 tags):
['ADJ', 'ADP', 'ADV', 'AUX', 'CCONJ', 'DET', 'NOUN', 'PART', 'PRON', 'PROPN', 'PUNCT', 'SPACE', '
```

```
print("\n--- Comparison of NLTK and spaCy POS Tags ---\n")

print("NLTK Unique Tags (fine-grained):")
print(unique_tags)
print(f"Number of NLTK tags: {len(unique_tags)}\n")

print("spaCy Unique Universal Tags:")
print(unique_universal_tags)
print(f"Number of spaCy universal tags: {len(unique_universal_tags)}\n")

# Example of mapping NLTK to Universal Tags (simplified)
print("Observations:")
print("- NLTK provides more granular tags (e.g., NNP, NNS, NN for nouns; VB, VBP, VBZ for verbs)
print("- spaCy's universal tags are broader categories (e.g., all nouns fall under NOUN, all ver
print("- Punctuation and spaces are also tagged differently (NLTK uses symbols like '(', ',', '.
```

```
--- Comparison of NLTK and spaCy POS Tags ---

NLTK Unique Tags (fine-grained):
["''", '(', ')', ',', '.', 'CC', 'DT', 'IN', 'JJ', 'NN', 'NNP', 'NNS', 'PRP', 'PRP$', 'RB', 'TO',
Number of NLTK tags: 21

spaCy Unique Universal Tags:
['ADJ', 'ADP', 'ADV', 'AUX', 'CCONJ', 'DET', 'NOUN', 'PART', 'PRON', 'PROPN', 'PUNCT', 'SPACE', '
Number of spaCy universal tags: 13

Observations:
- NLTK provides more granular tags (e.g., NNP, NNS, NN for nouns; VB, VBP, VBZ for verbs).
- spaCy's universal tags are broader categories (e.g., all nouns fall under NOUN, all verbs under
- Punctuation and spaces are also tagged differently (NLTK uses symbols like '(', ',', '.', etc.;
```

```
# Identify Nouns (Academic Concepts) and Verbs (Arguments) from NLTK tags
nltk_academic_concepts = [word for word, tag in pos_tags_nltk if tag.startswith('NN')]
nltk_arguments = [word for word, tag in pos_tags_nltk if tag.startswith('VB')]

print("--- NLTK Identified Nouns (Academic Concepts) ---")
print(nltk_academic_concepts)
print(f"Total Nouns: {len(nltk_academic_concepts)}\n")

print("--- NLTK Identified Verbs (Arguments) ---")
print(nltk_arguments)
print(f"Total Verbs: {len(nltk_arguments)}\n")

# Identify Nouns (Academic Concepts) and Verbs (Arguments) from spaCy universal tags
spacy_academic_concepts = [word for word, tag in pos_tags_spacy if tag in ['NOUN', 'PROPN']]
spacy_arguments = [word for word, tag in pos_tags_spacy if tag in ['VERB', 'AUX']]

print("--- spaCy Identified Nouns (Academic Concepts) ---")
print(spacy_academic_concepts)
print(f"Total Nouns: {len(spacy_academic_concepts)}\n")

print("--- spaCy Identified Verbs (Arguments) ---")
print(spacy_arguments)
print(f"Total Verbs: {len(spacy_arguments)}\n")
```

```
--- NLTK Identified Nouns (Academic Concepts) ---
['Language', 'Processing', 'NLP', 'field', 'intelligence', 'computers', 'language', 'modeling', '
Total Nouns: 26

--- NLTK Identified Verbs (Arguments) ---
```

```
['is', 'enables', 'understand', 'interpret', 'generate', 'combines', 'enable', 'process', "'under
Total Verbs: 11

--- spaCy Identified Nouns (Academic Concepts) ---
['Natural', 'Language', 'Processing', 'NLP', 'field', 'intelligence', 'computers', 'language', 'l
Total Nouns: 28

--- spaCy Identified Verbs (Arguments) ---
['is', 'enables', 'understand', 'interpret', 'generate', 'combines', 'based', 'enable', 'process'
Total Verbs: 11
```

```
from collections import Counter

# NLTK Frequencies
nltk_noun_freq = Counter(nltk_academic_concepts)
nltk_verb_freq = Counter(nltk_arguments)

print("--- NLTK Noun Frequencies (Top 10) ---")
for word, count in nltk_noun_freq.most_common(10):
    print(f"{word}: {count}")
print(f"Total unique NLTK nouns: {len(nltk_noun_freq)}\n")

print("--- NLTK Verb Frequencies (Top 10) ---")
for word, count in nltk_verb_freq.most_common(10):
    print(f"{word}: {count}")
print(f"Total unique NLTK verbs: {len(nltk_verb_freq)}\n")

# spaCy Frequencies
spacy_noun_freq = Counter(spacy_academic_concepts)
spacy_verb_freq = Counter(spacy_arguments)

print("--- spaCy Noun Frequencies (Top 10) ---")
for word, count in spacy_noun_freq.most_common(10):
    print(f"{word}: {count}")
print(f"Total unique spaCy nouns: {len(spacy_noun_freq)}\n")

print("--- spaCy Verb Frequencies (Top 10) ---")
for word, count in spacy_verb_freq.most_common(10):
    print(f"{word}: {count}")
print(f"Total unique spaCy verbs: {len(spacy_verb_freq)}\n")
```

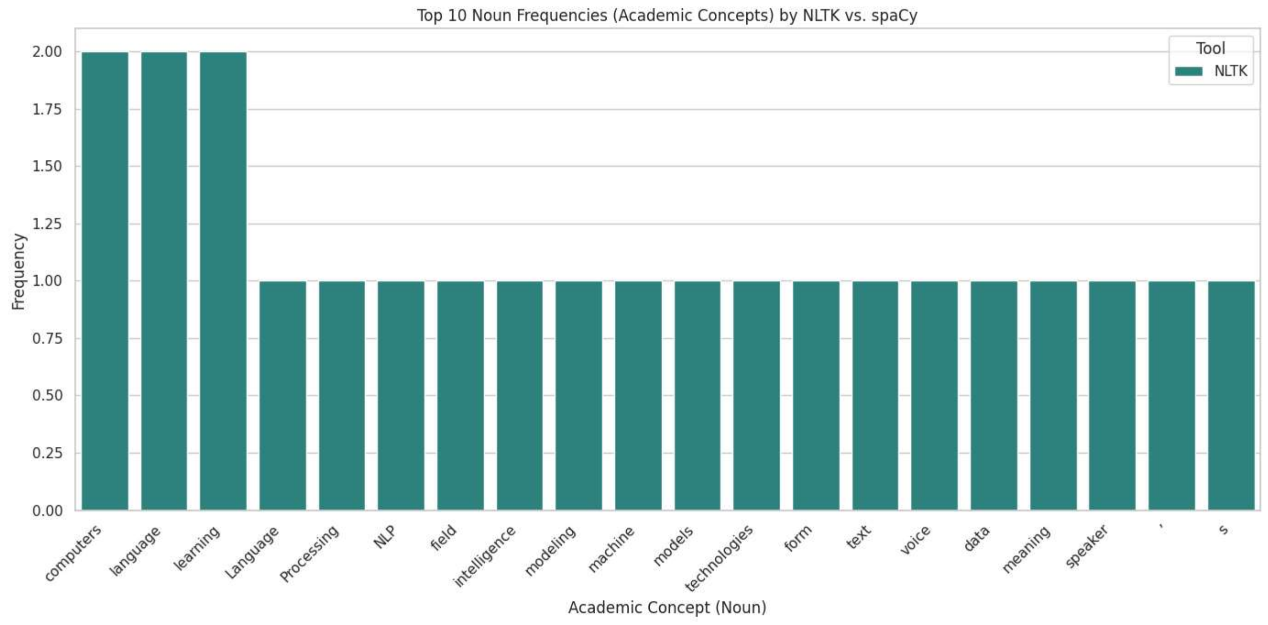
```
--- NLTK Noun Frequencies (Top 10) ---
computers: 2
language: 2
learning: 2
Language: 1
Processing: 1
NLP: 1
field: 1
intelligence: 1
modeling: 1
machine: 1
Total unique NLTK nouns: 23

--- NLTK Verb Frequencies (Top 10) ---
is: 1
enables: 1
understand: 1
interpret: 1
generate: 1
combines: 1
```

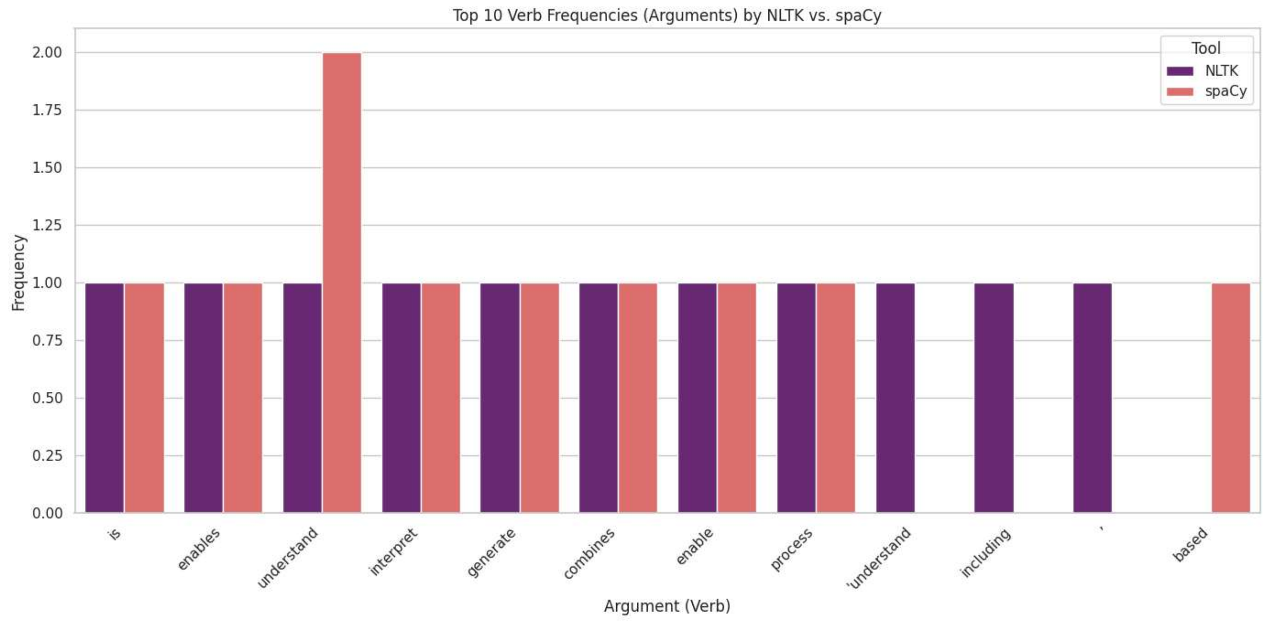
```
enable: 1
process: 1
'understand: 1
including: 1
Total unique NLTK verbs: 11

--- spaCy Noun Frequencies (Top 10) ---
language: 3
computers: 2
learning: 2
Natural: 1
Language: 1
Processing: 1
NLP: 1
field: 1
intelligence: 1
linguistics: 1
Total unique spaCy nouns: 24

--- spaCy Verb Frequencies (Top 10) ---
understand: 2
is: 1
enables: 1
interpret: 1
generate: 1
combines: 1
based: 1
enable: 1
process: 1
including: 1
Total unique spaCy verbs: 10
```

```python
import pandas as pd

# Create DataFrame for NLTK frequencies
nltk_freq_data = []
for word, count in nltk_noun_freq.items():
    nltk_freq_data.append({'Tool': 'NLTK', 'Type': 'Noun', 'Word': word, 'Frequency': count})
for word, count in nltk_verb_freq.items():
    nltk_freq_data.append({'Tool': 'NLTK', 'Type': 'Verb', 'Word': word, 'Frequency': count})
nltk_freq_df = pd.DataFrame(nltk_freq_data)

# Create DataFrame for spaCy frequencies
spacy_freq_data = []
for word, count in spacy_noun_freq.items():
    spacy_freq_data.append({'Tool': 'spaCy', 'Type': 'Noun', 'Word': word, 'Frequency': count})
for word, count in spacy_verb_freq.items():
    spacy_freq_data.append({'Tool': 'spaCy', 'Type': 'Verb', 'Word': word, 'Frequency': count})
spacy_freq_df = pd.DataFrame(spacy_freq_data)

# Combine both DataFrames
combined_freq_df = pd.concat([nltk_freq_df, spacy_freq_df], ignore_index=True)

print("--- NLTK Frequencies DataFrame ---")
display(nltk_freq_df.head())
print("\n--- spaCy Frequencies DataFrame ---")
display(spacy_freq_df.head())
print("\n--- Combined Frequencies DataFrame (First 10 Rows) ---")
display(combined_freq_df.head(10))
```

```
--- NLTK Frequencies DataFrame ---
   Tool  Type          Word  Frequency
0  NLTK  Noun      Language          1
1  NLTK  Noun    Processing          1
2  NLTK  Noun           NLP          1
3  NLTK  Noun         field          1
4  NLTK  Noun  intelligence          1

--- spaCy Frequencies DataFrame ---
    Tool  Type        Word  Frequency
0  spaCy  Noun     Natural          1
1  spaCy  Noun    Language          1
2  spaCy  Noun  Processing          1
3  spaCy  Noun         NLP          1
4  spaCy  Noun       field          1

--- Combined Frequencies DataFrame (First 10 Rows) ---
   Tool  Type          Word  Frequency
0  NLTK  Noun      Language          1
1  NLTK  Noun    Processing          1
2  NLTK  Noun           NLP          1
3  NLTK  Noun         field          1
4  NLTK  Noun  intelligence          1
5  NLTK  Noun     computers          2
6  NLTK  Noun      language          2
7  NLTK  Noun      modeling          1
8  NLTK  Noun       machine          1
9  NLTK  Noun      learning          2
```

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Set a style for better aesthetics
sns.set_theme(style="whitegrid")

# --- Visualize Noun Frequencies ---
noun_df = combined_freq_df[combined_freq_df['Type'] == 'Noun'].copy()
noun_df_sorted = noun_df.sort_values(by=['Tool', 'Frequency'], ascending=[True, False])

plt.figure(figsize=(14, 7))
sns.barplot(data=noun_df_sorted.head(20), x='Word', y='Frequency', hue='Tool', palette='viridis')
plt.title('Top 10 Noun Frequencies (Academic Concepts) by NLTK vs. spaCy')
plt.xlabel('Academic Concept (Noun)')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# --- Visualize Verb Frequencies ---
verb_df = combined_freq_df[combined_freq_df['Type'] == 'Verb'].copy()
verb_df_sorted = verb_df.sort_values(by=['Tool', 'Frequency'], ascending=[True, False])

plt.figure(figsize=(14, 7))
sns.barplot(data=verb_df_sorted.head(20), x='Word', y='Frequency', hue='Tool', palette='magma')
plt.title('Top 10 Verb Frequencies (Arguments) by NLTK vs. spaCy')
plt.xlabel('Argument (Verb)')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```


Top 10 Noun Frequencies (Academic Concepts) by NLTK vs. spaCy
Tool
NLTK
Frequency
computers
language
learning
Language
Processing
NLP
field
intelligence
modeling
machine
models
technologies
form
text
voice
data
meaning
speaker
'
s
Academic Concept (Noun)


Top 10 Verb Frequencies (Arguments) by NLTK vs. spaCy
Tool
NLTK
spaCy
Frequency
is
enables
understand
interpret
generate
combines
enable
process
'understand
including
'
based
Argument (Verb)