```
!pip install spacy pandas matplotlib seaborn wordcloud
!python -m spacy download en_core_web_sm
```

```
Requirement already satisfied: spacy in /usr/local/lib/python3.12/dist-packages (3.8.11)
Requirement already satisfied: pandas in /usr/local/lib/python3.12/dist-packages (2.2.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.12/dist-packages (3.10.0)
Requirement already satisfied: seaborn in /usr/local/lib/python3.12/dist-packages (0.13.2)
Requirement already satisfied: wordcloud in /usr/local/lib/python3.12/dist-packages (1.9.5)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.12/dist-packages (from spacy) (3.0.12)
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (1.0.5)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (1.0.15)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.0.13)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.12/dist-packages (from spacy) (3.0.12)
Requirement already satisfied: thinc<8.4.0,>=8.3.4 in /usr/local/lib/python3.12/dist-packages (from spacy) (8.3.10)
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.12/dist-packages (from spacy) (1.1.3)
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.5.2)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.0.10)
Requirement already satisfied: weasel<0.5.0,>=0.4.2 in /usr/local/lib/python3.12/dist-packages (from spacy) (0.4.3)
Requirement already satisfied: typer-slim<1.0.0,>=0.3.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (0.21.1)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (4.67.1)
Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.0.2)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.32.4)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.12/dist-packages (from spacy) (
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from spacy) (3.1.6)
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from spacy) (75.2.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (25.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas) (2025.3)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (4.61.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.4.9)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (11.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (3.3.1)
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!=1.8.1,
Requirement already satisfied: pydantic-core==2.41.4 in /usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!=1.8.1,<
Requirement already satisfied: typing-extensions>=4.14.1 in /usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!=1.8
Requirement already satisfied: typing-inspection>=0.4.2 in /usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!=1.8.
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.8.2->pandas) (1.1
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.1
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->spacy)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->s
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->s
Requirement already satisfied: blis<1.4.0,>=1.3.0 in /usr/local/lib/python3.12/dist-packages (from thinc<8.4.0,>=8.3.4->spacy
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.12/dist-packages (from thinc<8.4.0,>=8.3.4-
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.12/dist-packages (from typer-slim<1.0.0,>=0.3.0->spacy)
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/python3.12/dist-packages (from weasel<0.5.0,>=0.4
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/python3.12/dist-packages (from weasel<0.5.0,>=0.4.2
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->spacy) (3.0.3)
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from smart-open<8.0.0,>=5.2.1->weasel<0.5.0,
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_core_web_sm-3.8.0-py3-none-
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.8/12.8 MB 61.7 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.
```

```
import pandas as pd
import spacy
from collections import Counter
import matplotlib.pyplot as plt
import seaborn as sns
from spacy.matcher import Matcher
from wordcloud import WordCloud
```

```
import pandas as pd

df = pd.read_csv(
    "/content/arxiv_data.csv",
    engine="python",
    on_bad_lines="skip"
)

df.head()
```

| | titles | summaries | terms |
|---|---|---|---|
| **0** | Survey on Semantic Stereo Matching / Semantic ... | Stereo matching is one of the widely used tech... | ['cs.CV', 'cs.LG'] |
| **1** | FUTURE-AI: Guiding Principles and Consensus Re... | The recent advancements in artificial intellig... | ['cs.CV', 'cs.AI', 'cs.LG'] |
| **2** | Enforcing Mutual Consistency of Hard Regions f... | In this paper, we proposed a novel mutual cons... | ['cs.CV', 'cs.AI'] |

```
# Filter CS / AI related abstracts
cs_df = df[df['terms'].str.contains('cs|AI', case=False, na=False)]

# Keep only abstracts
abstracts = cs_df['summaries'].dropna().head(500)  # limit for faster processing

len(abstracts)
```

```
500
```

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

```
docs = list(nlp.pipe(abstracts))
```

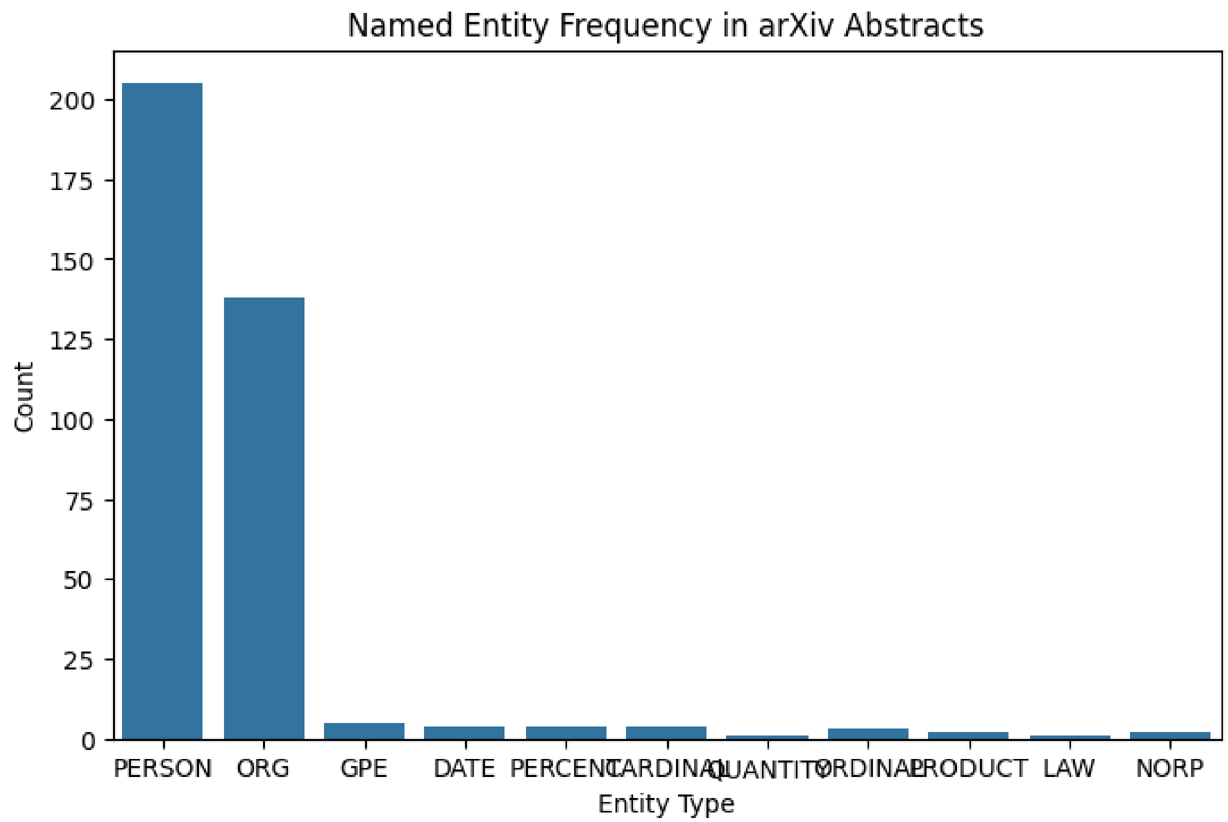
```
noun_phrases = []

for doc in docs:
    for chunk in doc.noun_chunks:
        if len(chunk.text) > 2:
            noun_phrases.append(chunk.text.lower())

np_freq = Counter(noun_phrases)
top_noun_phrases = np_freq.most_common(15)

top_noun_phrases
```

```
[('"[\'cs.cv', 201),
 ('"[\'cs.lg', 106),
 ('this paper', 10),
 ('this work', 4),
 ('the-art', 3),
 ("the radiologist's workload", 2),
 ('practice', 2),
 ('our method', 2),
 ('a low data regime', 1),
 ('a novel deep learning-based approach', 1),
 (' cocostuff', 1),
 ('cityscapes', 1),
 (' school', 1),
 ('recent years', 1),
 ('$\\textit{inextremis}$', 1)]
```

```
entities = []

for doc in docs:
    for ent in doc.ents:
        entities.append((ent.text, ent.label_))

entity_labels = [label for _, label in entities]
entity_freq = Counter(entity_labels)

entity_freq
```

```
Counter({'PERSON': 205,
         'ORG': 138,
         'GPE': 5,
         'DATE': 4,
         'PERCENT': 4,
         'CARDINAL': 4,
         'QUANTITY': 1,
         'ORDINAL': 3,
         'PRODUCT': 2,
         'LAW': 1,
         'NORP': 2})
```

```
plt.figure(figsize=(8,5))
sns.barplot(x=list(entity_freq.keys()), y=list(entity_freq.values()))
plt.title("Named Entity Frequency in arXiv Abstracts")
plt.xlabel("Entity Type")
plt.ylabel("Count")
plt.show()
```



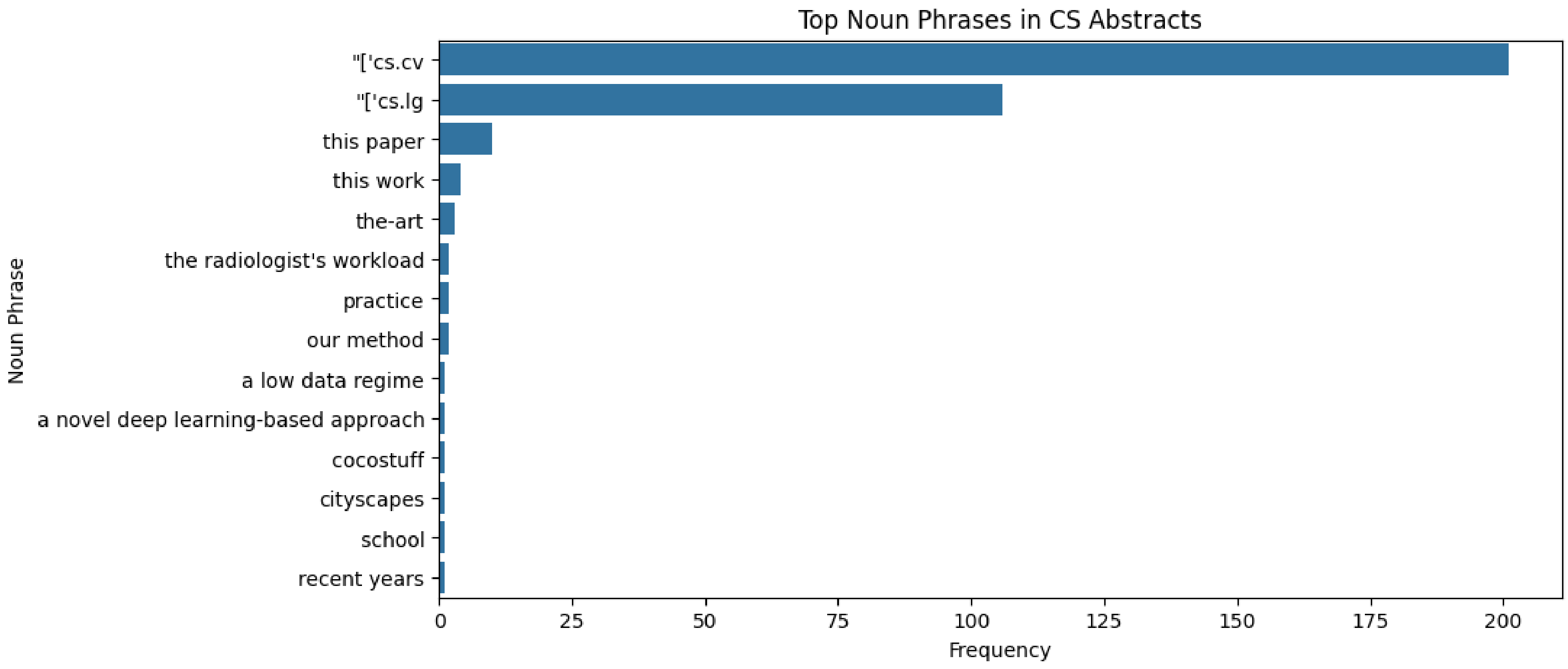
```
matcher = Matcher(nlp.vocab)

pattern1 = [{"LOWER": "neural"}, {"LOWER": "network"}]
pattern2 = [{"LOWER": "machine"}, {"LOWER": "learning"}]
pattern3 = [{"LOWER": "deep"}, {"LOWER": "learning"}]
pattern4 = [{"POS": "ADJ"}, {"POS": "NOUN"}, {"POS": "NOUN"}]

matcher.add("TECH_TERMS", [pattern1, pattern2, pattern3, pattern4])
```

```
matched_terms = []

for doc in docs:
    matches = matcher(doc)
    for match_id, start, end in matches:
        matched_terms.append(doc[start:end].text.lower())
```

```
Counter(matched_terms).most_common(15)
```

```
[('low data regime', 1),
 ('deep learning', 1),
 ('real pairwise data', 1),
 ('aerial imagery segmentation', 1),
 ('semantic segmentation models', 1),
 ('active shape model', 1),
 ('massive railway system', 1),
 ('semantic segmentation problem', 1),
 ('final solution quality', 1),
 ('unsupervised image segmentation', 1),
 ('short training times', 1),
 ('actual object positions', 1),
 ('identical network structure', 1),
 ('modal retrieval tasks', 1),
 ('explicit shape supervision', 1)]
```

```
filtered_noun_phrases = []
for phrase, count in top_noun_phrases:
    # Filter out phrases that contain LaTeX math delimiters or commands
    if '$' not in phrase and '\\' not in phrase:
        filtered_noun_phrases.append((phrase, count))

labels, values = zip(*filtered_noun_phrases)

plt.figure(figsize=(10,5))
sns.barplot(x=list(values), y=list(labels))
plt.title("Top Noun Phrases in CS Abstracts")
plt.xlabel("Frequency")
plt.ylabel("Noun Phrase")
plt.show()
```



1. spaCy's general-purpose NER model struggles with domain-specific entities such as algorithm names and mathematical terms.
2. Technical phrases are often split incorrectly due to complex syntax.
3. Domain-specific models (e.g., SciSpaCy) would improve accuracy for research papers.
4. Rule-based matchers help compensate but require manual effort.

This experiment demonstrates how spaCy can effectively process research abstracts to extract linguistic patterns and entities. However, for highly technical domains such as AI research, domain-adapted NLP models are recommended.