```python
# ===============================
# 1. Import Required Libraries
# ===============================
import pandas as pd
import spacy
from spacy.matcher import Matcher
from collections import Counter
import matplotlib.pyplot as plt
```

```python
 #2. Load spaCy Model
# ===============================
nlp = spacy.load("en_core_web_sm")
```

```python
df = pd.read_csv("/content/arxiv_data.csv", on_bad_lines='skip', engine='python')
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']
3	Parameter Decoupling Strategy for Semi-supervi...	Consistency training has proven to be an advan...	['cs.CV']
4	Background-Foreground Segmentation for Interio...	To ensure safety in automated driving, the cor...	['cs.CV', 'cs.LG']

Next steps: (Generate code with df)   (New interactive sheet)

```python
print("DataFrame columns:", df.columns.tolist())
```

```
DataFrame columns: ['titles', 'summaries', 'terms']
```

```python
# Remove empty abstracts
df = df.dropna(subset=["summaries"])
```

```python
# Take a subset for faster processing
texts = df["summaries"].head(200).tolist()
```

```python
# 4. Process Text Using spaCy Pipeline
# ===============================
docs = list(nlp.pipe(texts))
```

```python
# 5. Extract Frequent Noun Phrases
# ===============================
noun_phrases = []

for doc in docs:
    for chunk in doc.noun_chunks:
        noun_phrases.append(chunk.text.lower())

np_freq = Counter(noun_phrases)
top_noun_phrases = np_freq.most_common(10)

print("Top Noun Phrases:")
for phrase, count in top_noun_phrases:
    print(f"{phrase} : {count}")
```

```
Top Noun Phrases:
we : 540
which : 172
that : 144
it : 120
this paper : 74
the-art : 72
our method : 50
```

```
    image segmentation : 47
    this work : 47
    medical image segmentation : 37
```

```python
# 6. Extract Named Entities
# ===============================
entities = []

for doc in docs:
    for ent in doc.ents:
        if ent.label_ in ["ORG", "DATE", "PRODUCT", "GPE", "PERSON"]:
            entities.append(ent.label_)

entity_freq = Counter(entities)

print("\nNamed Entity Frequency:")
for ent, count in entity_freq.items():
    print(f"{ent} : {count}")
```

```
Named Entity Frequency:
DATE : 28
GPE : 44
ORG : 525
PERSON : 60
PRODUCT : 15
```

```python
# 7. spaCy Matcher for Technical Terms
# ===============================
matcher = Matcher(nlp.vocab)

# Define technical term patterns
pattern1 = [{"LOWER": "machine"}, {"LOWER": "learning"}]
pattern2 = [{"LOWER": "deep"}, {"LOWER": "learning"}]
pattern3 = [{"LOWER": "neural"}, {"LOWER": "network"}]
pattern4 = [{"LOWER": "support"}, {"LOWER": "vector"}, {"LOWER": "machine"}]

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

tech_terms = []

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

tech_term_freq = Counter(tech_terms)

print("\nTop Technical Terms:")
for term, count in tech_term_freq.most_common(10):
    print(f"{term} : {count}")
```
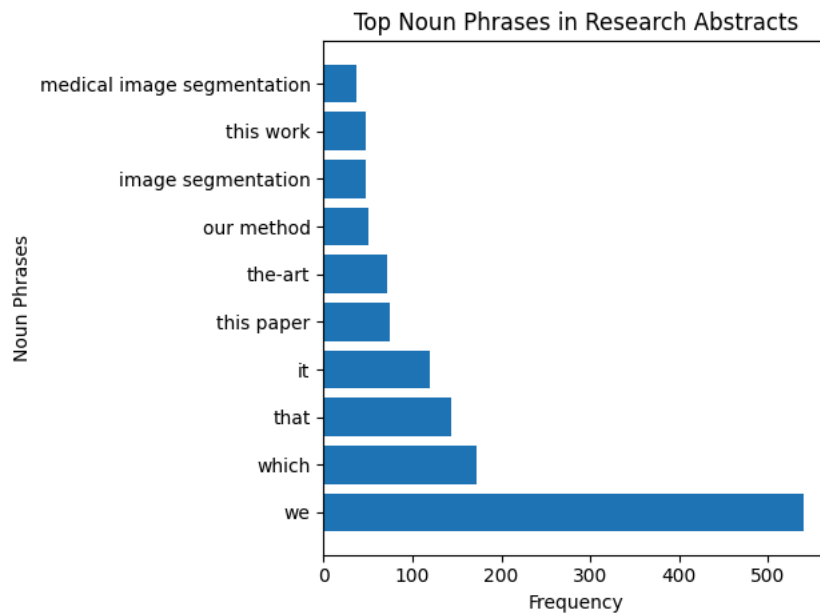
```
Top Technical Terms:
deep learning : 45
neural network : 20
machine learning : 17
```

```python
# 8. Visualization: Top Noun Phrases
# ===============================
labels_np, values_np = zip(*top_noun_phrases)

plt.figure()
plt.barh(labels_np, values_np)
plt.xlabel("Frequency")
plt.ylabel("Noun Phrases")
plt.title("Top Noun Phrases in Research Abstracts")
plt.tight_layout()
plt.show()
```

Top Noun Phrases in Research Abstracts


```
# 9. Visualization: Entity Frequency
# =================================
labels_ent, values_ent = zip(*entity_freq.items())

plt.figure()
plt.bar(labels_ent, values_ent)
plt.xlabel("Entity Type")
plt.ylabel("Count")
plt.title("Named Entity Frequency")
plt.tight_layout()
plt.show()
```

Named Entity Frequency