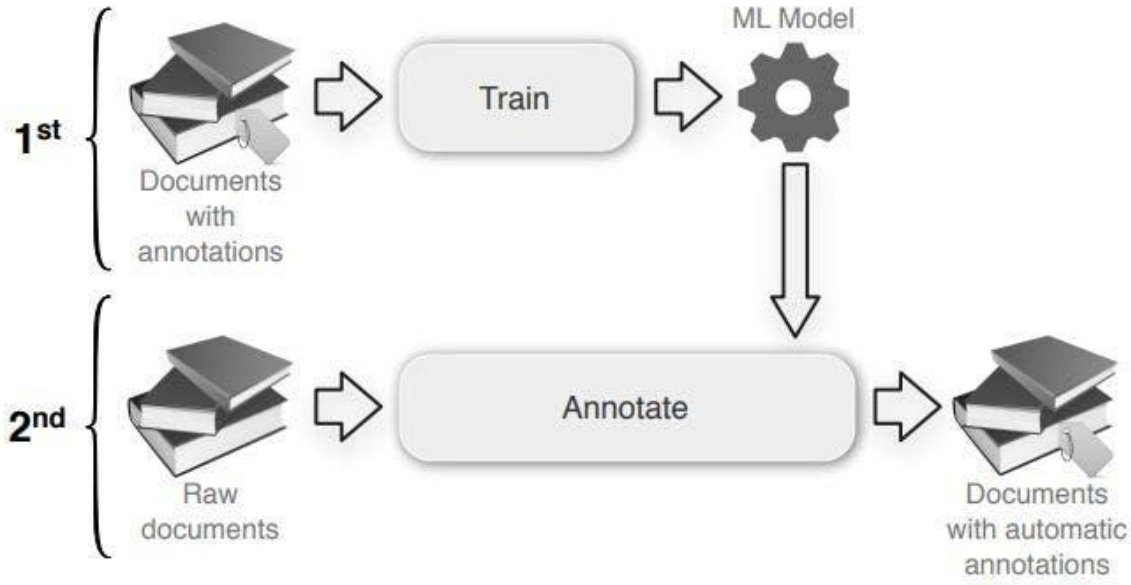
Introduction

Named Entity Recognition (NER) is a crucial NLP technique that allows machines to **automatically identify and categorize important elements in text**, such as names, dates, organizations, and locations. From my perspective, NER acts like a smart highlighter—it transforms unstructured text into structured insights, making it easier to analyze and act upon. Using advanced models like **spaCy's transformer-based pipeline**, we can achieve high accuracy in recognizing entities while also providing flexibility to handle different text types and domains. This makes NER invaluable for tasks like information extraction, data analysis, and automated content understanding.



Installing PIP & Importing The Files

```
# Install required packages if not already installed
# pip install spacy spacy-transformers pandas matplotlib
# python -m spacy download en_core_web_trf

import spacy
from spacy import displacy
import pandas as pd
from collections import Counter
import matplotlib.pyplot as plt
```

```
[3] pip install torch
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from torch) (3.19.1)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.12/dist-packages (from torch) (4.14.1)
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch) (75.2.0)
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch) (1.13.3)
Requirement already satisfied: networkx in /usr/local/lib/python3.12/dist-packages (from torch) (3.5)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec in /usr/local/lib/python3.12/dist-packages (from torch) (2025.3.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.80)
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch) (9.10.2.21)
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.4.1)
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch) (11.3.0.4)
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch) (10.3.7.77)
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch) (11.7.1.2)
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch) (12.5.4.2)
Requirement already satisfied: nvidia-cusparselt-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch) (0.7.1)
Requirement already satisfied: nvidia-nccl-cu12==2.27.3 in /usr/local/lib/python3.12/dist-packages (from torch) (2.27.3)
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.85)
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch) (1.11.1.6)
Requirement already satisfied: triton==3.4.0 in /usr/local/lib/python3.12/dist-packages (from torch) (3.4.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from sympy>=1.13.3->torch) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->torch) (3.0.2)
```

Loading NLP Model /Input Text/Process Text

Mohamed Raaiz

```python
# Step 1: Load NLP Model Safely
def load_model():
    try:
        # Try loading transformer-based model first
        nlp = spacy.load("en_core_web_trf")
        print("Loaded transformer-based model: en_core_web_trf")
    except (OSError, ValueError) as e:
        print("Transformer model not available. Falling back to small model.")
        print("Error:", e)
        try:
            nlp = spacy.load("en_core_web_sm")
            print("Loaded small English model: en_core_web_sm")
        except OSError:
            print("Small model not found. Downloading...")
            import subprocess
            subprocess.run(["python", "-m", "spacy", "download", "en_core_web_sm"])
            nlp = spacy.load("en_core_web_sm")
    return nlp
nlp = load_model()
```

```
Transformer model not available. Falling back to small model.
Error: [E002] Can't find factory for 'curated_transformer' for language English (en). This usually h

Available factories: merge_noun_chunks, merge_entities, merge_subtokens, en.lemmatizer
Loaded small English model: en_core_web_sm
```

```python
[12] # Step 2: Input Text
     # ----------------------------
     text = """
     Apple Inc. announced a new iPhone on September 12, 2025 in Cupertino, California.
     Elon Musk visited New York City on 15th August 2025 to discuss SpaceX partnerships.
     Microsoft Corp. and Google LLC signed a contract on January 10, 2025.
     """
```

```python
[13] # Step 3: Process Text

     doc = nlp(text)
```

Extract Entities & Count Entity Types

Mohamed Raaiz

```python
# Step 4: Extract Entities
entities = [(ent.text, ent.label_) for ent in doc.ents]
if not entities:
    print("No entities found.")
else:
    df_entities = pd.DataFrame(entities, columns=["Entity", "Type"])
    print("Extracted Entities:")
    print(df_entities)
```

```
Extracted Entities:
                Entity      Type
0           Apple Inc.       ORG
1   September 12, 2025      DATE
2            Cupertino       GPE
3           California       GPE
4            Elon Musk    PERSON
5         New York City       GPE
6                  15th   ORDINAL
7          August 2025      DATE
8       Microsoft Corp.       ORG
9          Google LLC        ORG
10   January 10, 2025      DATE
```

```python
[15] # Step 5: Count Entity Types
entity_counter = Counter([ent.label_ for ent in doc.ents])
print("\nEntity Counts:")
print(entity_counter)
```

```
Entity Counts:
Counter({'ORG': 3, 'DATE': 3, 'GPE': 3, 'PERSON': 1, 'ORDINAL': 1})
```

Visualize the entities

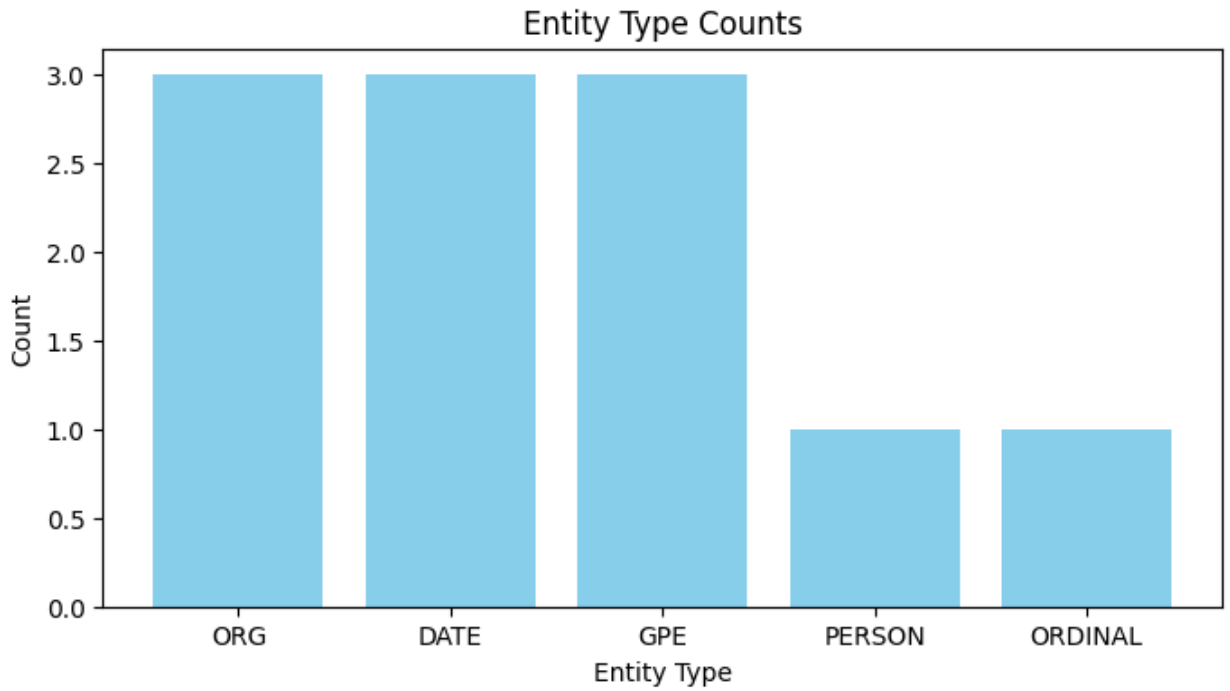
```python
# Step 6: Visualize Entities

# Inline visualization (Jupyter Notebook)
try:
    displacy.render(doc, style="ent", jupyter=True)
except:
    print("Displacy render works only in Jupyter Notebook.")
```

Apple Inc. ORG announced a new iPhone on September 12, 2025 DATE in Cupertino GPE , California GPE .
Elon Musk PERSON visited New York City GPE on 15th ORDINAL August 2025 DATE to discuss SpaceX partnerships.
Microsoft Corp. ORG and Google LLC ORG signed a contract on January 10, 2025 DATE .

```python
[17] # Bar chart
plt.figure(figsize=(8, 4))
plt.bar(entity_counter.keys(), entity_counter.values(), color="skyblue")
plt.title("Entity Type Counts")
plt.xlabel("Entity Type")
plt.ylabel("Count")
plt.show()
```

Mohamed Raaiz




```
# Step 7: Filter by Specific Types
people = [ent.text for ent in doc.ents if ent.label_ == "PERSON"]
dates = [ent.text for ent in doc.ents if ent.label_ == "DATE"]
companies = [ent.text for ent in doc.ents if ent.label_ in ["ORG", "COMPANY"]]

print("\nPeople:", people)
print("Dates:", dates)
print("Companies:", companies)
```

```
People: ['Elon Musk']
Dates: ['September 12, 2025', 'August 2025', 'January 10, 2025']
Companies: ['Apple Inc.', 'Microsoft Corp.', 'Google LLC']
```

```
[19] # Step 8: Export to CSV
try:
    df_entities.to_csv("ner_output.csv", index=False)
    print("Entities exported to ner_output.csv")
except Exception as e:
    print("Error saving CSV:", e)
```

```
Entities exported to ner_output.csv
```

Mohamed Raaiz

Conclusion

From my perspective, NER is not just about labeling words—it's about unlocking the hidden structure within text. By leveraging a robust NLP pipeline, we can quickly extract actionable information, visualize patterns, and organize data efficiently. While transformer models offer superior accuracy, it's important to implement fallback mechanisms for reliability across different environments. Overall, NER provides a bridge between raw textual data and meaningful insights, empowering applications that range from business intelligence to AI-driven analytics.

Mohamed Raaiz