

```
# Cài đặt spaCy
!pip install -U spacy

# Tải về mô hình tiếng Anh
!python -m spacy download en_core_web_md

Requirement already satisfied: spacy in
/usr/local/lib/python3.12/dist-packages (3.8.11)
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) (0.20.0)
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) (2.12.3)
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: annotated-types>=0.6.0 in
/usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!!=1.8.1,<3.0.0,>=1.7.4->spacy) (0.7.0)
Requirement already satisfied: pydantic-core==2.41.4 in
/usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!!=1.8.1,<3.0.0,>=1.7.4->spacy) (2.41.4)
```

```
Requirement already satisfied: typing-extensions>=4.14.1 in
/usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!-1.8.1,<3.0.0,>=1.7.4->spacy) (4.15.0)
Requirement already satisfied: typing-inspection>=0.4.2 in
/usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!-1.8.1,<3.0.0,>=1.7.4->spacy) (0.4.2)
Requirement already satisfied: charset_normalizer<4,>=2 in
/usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->spacy) (3.4.4)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->spacy) (3.11)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->spacy) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->spacy) (2025.11.12)
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) (1.3.3)
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->spacy) (0.1.5)
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) (8.3.1)
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in
/usr/local/lib/python3.12/dist-packages (from weasel<0.5.0,>=0.4.2->spacy) (0.23.0)
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->spacy) (7.5.0)
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,>=0.4.2->spacy) (2.0.1)
Collecting en-core-web-md==3.8.0
  Downloading
    https://github.com/explosion/spacy-models/releases/download/en_core_web_md-3.8.0/en_core_web_md-3.8.0-py3-none-any.whl (33.5 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 33.5/33.5 MB 13.1 MB/s eta
0:00:0000:0100:01
d
Successfully installed en-core-web-md-3.8.0
✓ Download and installation successful
You can now load the package via spacy.load('en_core_web_md')
△ 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 spacy  
from spacy import displacy
```

Phần 2: Phân tích câu và Trực quan hóa

Trực quan hóa là cách tốt nhất để bắt đầu hiểu về cây phụ thuộc. spaCy cung cấp một công cụ tuyệt vời tên là displaCy.

2.1. Tải mô hình và phân tích câu

```
nlp = spacy.load("en_core_web_md")  
  
# Câu ví dụ  
text = "The quick brown fox jumps over the lazy dog."  
  
# Phân tích câu với pipeline cu'a spaCy  
doc = nlp(text)  
  
print(f"'TEXT':<12} | {'DEP':<10} | {'HEAD TEXT':<12} | {'HEAD  
POS':<8} | {'CHILDREN'}")  
print("-" * 70)  
  
for token in doc:  
    children = [child.text for child in token.children]  
    print(f"{token.text:<12} | {token.dep_:<10} |  
{token.head.text:<12} | {token.head.pos_:<8} | {children}")
```

TEXT	DEP	HEAD TEXT	HEAD POS	CHILDREN
The	det	fox	NOUN	[]
quick	amod	fox	NOUN	[]
brown	amod	fox	NOUN	[]
fox	nsubj	jumps	VERB	['The', 'quick', 'brown']
jumps	ROOT	jumps	VERB	['fox', 'over', '.']
over	prep	jumps	VERB	['dog']
the	det	dog	NOUN	[]
lazy	amod	dog	NOUN	[]
dog	pobj	over	ADP	['the', 'lazy']
.	punct	jumps	VERB	[]

```
# Tùy chọn để hiển thị trong trình duyệt  
# options = {"compact": True, "color": "blue", "font": "Source Sans"}
```

```

Pro"}
# Kho'i chạy server tại http://127.0.0.1:5000
# Bạn có thể truy cập địa chỉ này trên trình duyệt để xem cây phụ
# Nhấn Ctrl+C trong terminal để dừng server
# displacy.serve(doc, style="dep")

# Hoặc hiển thị trong notebook
displacy.render(doc, style="dep", jupyter=True, options={"distance": 100})

<IPython.core.display.HTML object>

```

- Từ jumps là gốc (ROOT) của câu
- jumps có những từ phụ thuộc (dependent) là fox, over. Các quan hệ đó là: nsubj, prep
- fox là head của những từ: brown, quick, The

Phần 3: Truy cập các thành phần trong cây phụ thuộc

Trực quan hóa rất hữu ích, nhưng sức mạnh thực sự đến từ việc truy cập cây phụ thuộc theo chương trình. Mỗi Token trong đối tượng Doc của spaCy chứa đầy đủ thông tin về vị trí của nó trong cây.

TEXT	DEP	HEAD TEXT	HEAD POS	CHILDREN
Apple	nsubj	looking	VERB	[]
is	aux	looking	VERB	[]
looking	ROOT	looking	VERB	['Apple', 'is' , 'at']
at	prep	looking	VERB	['buying']
buying	pcomp	at	ADP	['startup']
U.K.	compound	startup	NOUN	[]
startup	dobj	buying	VERB	['U.K.', 'for']
for	prep	startup	NOUN	['billion']
\$	quantmod	billion	NUM	[]
1	compound	billion	NUM	[]
billion	pobj	for	ADP	['\$', '1']

Giải thích các thuộc tính:

- `token.text`: Văn bản của token.
- `token.dep_`: Nhãn quan hệ phụ thuộc của token này với head của nó.
- `token.head.text`: Văn bản của token head.
- `token.head.pos_`: Part-of-Speech tag của token head.
- `token.children`: Một iterator chứa các token con (dependent) của token hiện tại.

Phần 4: Duyệt cây phụ thuộc để trích xuất thông tin

4.1. Bài toán: Tìm chủ ngữ và tân ngữ của một động từ

```
text = "The cat chased the mouse and the dog watched them."
doc = nlp(text)
```

```
for token in doc:
    if token.pos_ == "VERB":
        verb = token.text
        subject = ""
        obj = ""

        for child in token.children:
            if child.dep_ == "nsubj":
                subject = child.text
            if child.dep_ == "dobj":
                obj = child.text

        if subject and obj:
            print(f"Found Triplet: ({subject}, {verb}, {obj})")
```

```
Found Triplet: (cat, chased, mouse)
Found Triplet: (dog, watched, them)
```

4.2. Bài toán: Tìm các tính từ bổ nghĩa cho một danh từ

```
text = "The big, fluffy white cat is sleeping on the warm mat."
doc = nlp(text)

for token in doc:
    if token.pos_ == "NOUN":
        adjectives = []

        for child in token.children:
            if child.dep_ == "amod":
                adjectives.append(child.text)

        if adjectives:
            print(f"Danh từ '{token.text}' được bô' nghĩa bởi các tính
từ: {adjectives}")
```

```
Danh từ 'cat' được bô'nghĩa bởi các tính từ: ['big', 'fluffy',  
'white']  
Danh từ 'mat' được bô'nghĩa bởi các tính từ: ['warm']
```

Phần 5: Bài tập tự luyện

Bài 1: Tìm động từ chính của câu

Động từ chính của câu thường có quan hệ ROOT. Viết một hàm `find_main_verb(doc)` nhận vào một đối tượng Doc của spaCy và trả về Token là động từ chính.

```
def find_main_verb(doc):  
    for token in doc:  
        if token.dep_ == "ROOT" and token.pos_ == "VERB":  
            return token  
    return None  
  
# Test hàm  
test_sentence = "The student studied hard and passed the exam."  
doc_test = nlp(test_sentence)  
main_verb = find_main_verb(doc_test)  
if main_verb:  
    print(f"Main verb: {main_verb.text}")  
else:  
    print("Not found main verb")  
  
Main verb: studied
```

Bài 2: Trích xuất các cụm danh từ (Noun Chunks)

spaCy đã có sẵn thuộc tính `.noun_chunks` để trích xuất các cụm danh từ. Tuy nhiên, hãy thử tự viết một hàm để làm điều tương tự.

Gợi ý: Một cụm danh từ đơn giản là một danh từ và tất cả các từ bổ nghĩa cho nó (như det, amod, compound). Bạn có thể bắt đầu từ một danh từ và duyệt ngược lên head hoặc duyệt xuống các children của nó.

```
def extract_noun_chunks(doc):  
    noun_chunks = []  
  
    for token in doc:  
        if token.pos_ == "NOUN":  
            chunk_tokens = []  
  
            for child in token.children:  
                if child.dep_ in ["det", "amod", "compound"]:
```

```

        chunk_tokens.append(child)

    chunk_tokens.append(token)
    chunk_tokens.sort(key=lambda t: t.i)

    chunk_text = " ".join([t.text for t in chunk_tokens])
    noun_chunks.append(chunk_text)

return noun_chunks

test_sentence = "The big, fluffy white cat is sleeping on the warm
mat."
doc_test = nlp(test_sentence)
chunks = extract_noun_chunks(doc_test)
print("The noun chunks:")
for chunk in chunks:
    print(f" - {chunk}")

print("\nResult from spaCy noun_chunks:")
for chunk in doc_test.noun_chunks:
    print(f" - {chunk.text}")

```

The noun chunks:

- The big fluffy white cat
- the warm mat

Result from spaCy noun_chunks:

- The big, fluffy white cat
- the warm mat

Bài 3: Tìm đường đi ngắn nhất trong cây

Viết một hàm `get_path_to_root(token)` để tìm đường đi từ một token bất kỳ lên đến gốc (ROOT) của cây. Hàm nên trả về một danh sách các token trên đường đi.

```

def get_path_to_root(token):
    path = [token]
    current = token

    while current.head != current:
        current = current.head
        path.append(current)
        if current.dep_ == "ROOT":
            break

    return path

test_sentence = "The quick brown fox jumps over the lazy dog."
doc_test = nlp(test_sentence)

```

```
dog_token = None
for token in doc_test:
    if token.text == "dog":
        dog_token = token
        break

if dog_token:
    path = get_path_to_root(dog_token)
    print(f"Path from '{dog_token.text}' upto ROOT:")
    for i, token in enumerate(path):
        print(f" {i+1}. {token.text} ({token.dep_})")
```

Path from 'dog' upto ROOT:

1. dog (pobj)
2. over (prep)
3. jumps (ROOT)