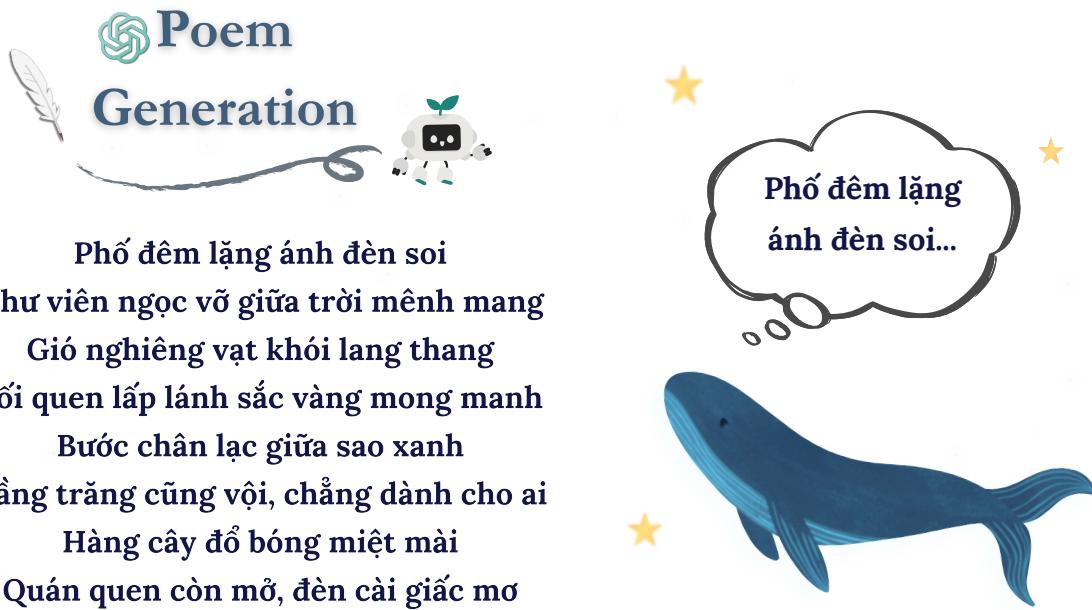


AI VIET NAM – AI COURSE 2024  
Project: Poem Generation

Dinh-Thang Duong, Yen-Linh Vu, Quang-Vinh Dinh

## I. Giới thiệu

Ngôn từ vốn mang vẻ đẹp của tri thức, và khi đi vào thơ ca, chúng mở ra một thế giới đầy tính nghệ thuật. Từ lâu, việc cảm thụ nghệ thuật, tư duy và sáng tạo được xem là đặc quyền của con người. Nhưng sẽ như thế nào nếu các mô hình học máy có thể nắm bắt được vẻ đẹp ấy, tự mình dệt nên những vần thơ mang nhịp điệu và ý nghĩa? Bài toán **Text Generation** đã mở ra khả năng cho máy tính không chỉ hiểu mà còn tạo ra ngôn ngữ một cách có ý thức ngữ cảnh. Đặc biệt, các mô hình như **ChatGPT** đã chứng minh khả năng sinh văn bản với độ trôi chảy và sáng tạo cao. Nhưng khi áp dụng vào sinh thơ, thách thức không chỉ nằm ở việc tạo ra câu từ hợp lý, mà còn phải đảm bảo vần điệu, nhịp điệu và sắc thái ngữ cảnh.



Hình 1: Bài toán sinh thơ tiếng Việt (thơ trong hình được tạo từ mô hình sinh thơ).

Thơ ca Việt Nam có nhiều thể loại phong phú, mỗi thể thơ mang đặc trưng về nhịp điệu, số chữ trong câu, cách gieo vần và ý nghĩa biểu đạt. Trong số đó, **thơ ngũ ngôn** hay thơ năm chữ là một thể thơ đặc biệt phô biến, đơn giản nhưng vẫn giữ được sự hài hòa về âm điệu. Mỗi dòng

gồm 5 âm tiết, thường được chia nhịp 2/3 hoặc 3/2 để tạo sự uyển chuyển khi đọc. Cách giao vần linh hoạt, thường là vần liền (AABB) hoặc vần cách (ABAB), giúp bài thơ có tính nhạc cao, phù hợp với việc diễn đạt tình cảm, thiên nhiên, hoặc triết lý nhân sinh.

Ví dụ về một đoạn thơ năm chữ sử dụng vần cách (ABAB):

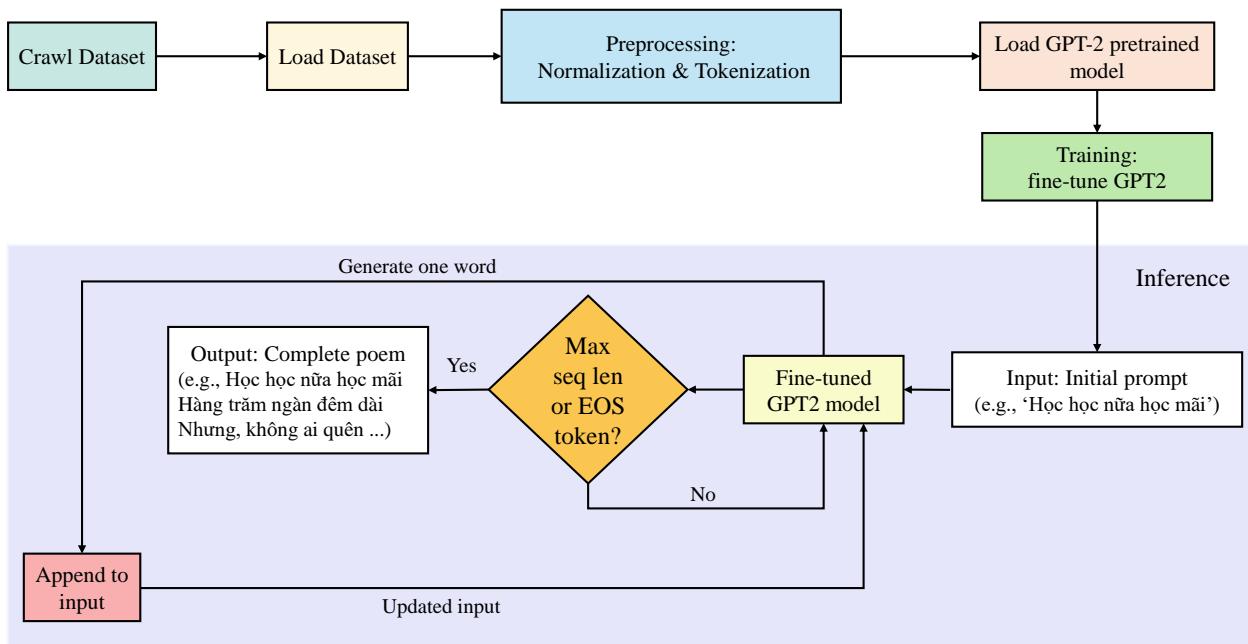
*Gió lồng từng khóm trúc  
Lá rì rào reo ca  
Dòng sông trôi ánh bạc  
Lững lờ con thuyền qua.*

— Tố Hữu, "Tiếng ru" (1956)

Trong project này, chúng ta sẽ cùng triển khai một chương trình sử dụng mô hình Text Generation với chủ đề sinh thơ Tiếng Việt, chủ yếu là thể thơ năm chữ dựa vào một từ tiếng Việt đầu vào từ người dùng. **Như vậy, Input/Output của chương trình là:**

- **Input:** Một chuỗi gồm các kí tự mở đầu cho bài thơ.
- **Output:** Bài thơ hoàn chỉnh.

Dựa theo nội dung được mô tả ở trên, ta sẽ có một pipeline với các bước thực hiện trong chương trình tổng thể như sau:



Hình 2: Pipeline mô tả bài toán sinh thơ tiếng Việt bằng GPT-2.

1. **Crawl Dataset (Thu thập dữ liệu):** Dữ liệu đầu vào là các bài thơ Tiếng Việt, được thu thập từ nhiều nguồn khác nhau. Quá trình này bao gồm:

- Crawl dữ liệu từ các trang web hoặc tập dữ liệu có sẵn.
- Lưu trữ và kiểm tra dữ liệu để đảm bảo chất lượng.

2. **Load and Preprocessing Data (Tải và tiền xử lý dữ liệu):** Sau khi thu thập dữ liệu, cần làm sạch và chuyển đổi dữ liệu về định dạng phù hợp. Các bước chính gồm:

- Chuẩn hóa văn bản: loại bỏ ký tự đặc biệt, xử lý khoảng trắng, chuẩn hóa dấu câu.
- Tokenization: chuyển văn bản thành các token để mô hình GPT-2 có thể xử lý.
- Tạo tập huấn luyện từ dữ liệu đã xử lý.

3. **Training (Huấn luyện mô hình):** Chúng ta tải mô hình GPT-2 đã được tiền huấn luyện và thực hiện fine-tuning với tập dữ liệu thơ tiếng Việt:

- Nạp mô hình GPT-2 pre-trained.
- Dào tạo mô hình trên tập dữ liệu thơ.
- Lưu mô hình đã fine-tune để sử dụng cho quá trình suy luận.

4. **Inference (Thực hiện dự đoán):** Sau khi mô hình được huấn luyện, quá trình sinh thơ được thực hiện như sau:

- Người dùng nhập vào một đoạn prompt ban đầu (ví dụ: “Học học nữa học mãi”).
- Prompt được token hóa và đưa vào mô hình GPT-2 đã fine-tune.
- Mô hình sinh ra từng token một theo cơ chế dự đoán từ tiếp theo.
- Token mới sinh ra được nối vào đầu vào và tiếp tục đưa vào mô hình.
- Quá trình lặp lại cho đến khi đạt độ dài tối đa hoặc gặp token kết thúc (EOS token).
- Kết quả cuối cùng là bài thơ hoàn chỉnh.

Pipeline này đảm bảo quá trình sinh thơ diễn ra tự động, từ thu thập dữ liệu đến tạo ra bài thơ chất lượng cao.

## II. Cài đặt chương trình

Trong phần này, chúng ta sẽ thực hiện hai giai đoạn chính của project để hoàn thiện được mô hình yêu cầu, bao gồm: Thu thập dữ liệu và Xây dựng mô hình. **Nội dung cụ thể như sau:**

### II.1. Thu thập dữ liệu

Để huấn luyện được mô hình với Input/Output theo đúng như yêu cầu đã đề ra ở phần trước, chúng ta cần thu thập vào xây dựng một bộ dữ liệu theo đúng mô tả. Đối với dữ liệu thơ, có rất nhiều trang web tổng hợp các bài thơ của Việt Nam cũng như thế giới. Tuy nhiên ở project này, ta sẽ thu thập các văn thơ ngũ ngôn trên trang web [thivien.net](http://thivien.net), một trang web lớn chuyên tổng hợp các văn thơ gồm đủ các thể loại của Việt Nam.

**\*Note:** Các bạn có thể tải bộ dữ liệu đã được thu thập sẵn tại phần IV. và bỏ qua bước này.

The screenshot shows the homepage of thivien.net. At the top, there is a header with social media icons (Facebook, Twitter, Pinterest, Tumblr) and a follow button. Below the header, a search bar displays 'Q. 93.054 bài thơ, 4.760 tác/dịch giả [Alt+3]'. To the right of the search bar is a 'Lọc' (Filter) button. Further right are navigation links for 'TÁC GIẢ', 'THƠ', 'THAM GIA', 'KHÁC', and a 'Đăng nhập' (Login) button. The main content area has two columns. The left column is titled 'Nội dung chính' and lists statistics: 'Thư viện thơ: 93.054 tác phẩm của 4.760 tác giả từ 104 nước, trong đó - 63.269 bài tiếng Việt - 17.207 bài chữ Hán - 12.578 bài ngôn ngữ khác', 'Thơ thành viên: 38.048 bài của 1.520 tác giả', 'Diễn đàn: 3.558 chủ đề', and 'Nội dung chi tiết theo các chuyên mục...'. Below this is a section titled 'Trong 24h qua, có...' with a link to 'HOT'. The right column is titled 'Trích điểm' and shows several snippets of poems in Vietnamese, such as 'Từ đạo lối về tôi khuất néo', 'Em còn nghẹng nón thận thùng che?', 'Đường xa có thấy lòng hiu quạnh', 'Dù nắng ngoài kia rực rỡ hè...', and a note '-- Vật nặng sân trường tôi bỏ lại (Nhược Thu)'. At the bottom of the right column is a section titled 'Tác giả mới' with links to profiles of Marius Chelaru (Rumania), Eric Patrick Clapton (Anh), and Alain Delorme (Pháp).

Hình 3: Trang chủ [thivien.net](http://thivien.net).

Có rất nhiều thư viện Python giúp ta có thể tương tác và trích xuất thông tin từ trang web một cách dễ dàng. Song ở project này, ta sẽ dùng thư viện [Selenium](#) để thực hiện việc thu thập dữ liệu trên Google Colab. **Các bước thực hiện như sau:**

#### II.1.1. Tải thư viện Selenium

Với môi trường máy tính cá nhân, ta đơn giản cài đặt bằng dòng lệnh `pip install selenium webdriver_manager`. Tuy nhiên với môi trường Google Colab, ta sẽ có cách cài đặt phức tạp hơn (chi tiết các bạn coi tại [đây](#)), các bạn hãy copy và chạy đoạn code bên dưới trong Colab:

```

1 %%shell
2 # Ubuntu no longer distributes chromium-browser outside of snap
3 #

```

```

4 # Proposed solution: https://askubuntu.com/questions/1204571/how-to-install-
      chromium-without-snap
5
6 # Add debian buster
7 cat > /etc/apt/sources.list.d/debian.list << "EOF"
8 deb [arch=amd64 signed-by=/usr/share/keyrings/debian-buster.gpg] http://deb.
      debian.org/debian buster main
9 deb [arch=amd64 signed-by=/usr/share/keyrings/debian-buster-updates.gpg] http
      ://deb.debian.org/debian buster-updates
      main
10 deb [arch=amd64 signed-by=/usr/share/keyrings/debian-security-buster.gpg] http
      ://deb.debian.org/debian-security
      buster/updates main
11 EOF
12
13 # Add keys
14 apt-key adv --keyserver keyserver.ubuntu.com --recv-keys DCC9EFBF77E11517
15 apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 648ACFD622F3D138
16 apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 112695A0E562B32A
17
18 apt-key export 77E11517 | gpg --dearmour -o /usr/share/keyrings/debian-buster.
      gpg
19 apt-key export 22F3D138 | gpg --dearmour -o /usr/share/keyrings/debian-buster-
      updates.gpg
20 apt-key export E562B32A | gpg --dearmour -o /usr/share/keyrings/debian-
      security-buster.gpg
21
22 # Prefer debian repo for chromium* packages only
23 # Note the double-blank lines between entries
24 cat > /etc/apt/preferences.d/chromium.pref << "EOF"
25 Package: *
26 Pin: release a=eoan
27 Pin-Priority: 500
28
29
30 Package: *
31 Pin: origin "deb.debian.org"
32 Pin-Priority: 300
33
34
35 Package: chromium*
36 Pin: origin "deb.debian.org"
37 Pin-Priority: 700
38 EOF
39
40 # Install chromium and chromium-driver
41 apt-get update
42 apt-get install chromium chromium-driver
43
44 # Install selenium
45 pip install selenium

```

## II.1.2. Import các thư viện cần thiết

```

1 import pandas as pd
2 import os
3 import requests
4 import time
5 import random
6
7 from tqdm import tqdm
8 from selenium import webdriver
9 from selenium.webdriver.chrome.service import Service
10 from selenium.webdriver.common.by import By
11 from selenium.webdriver.support.ui import WebDriverWait
12 from selenium.webdriver.support import expected_conditions as EC

```

### II.1.3. Khởi tạo Selenium driver

Với selenium, ta có thể hiểu đơn giản rằng việc truy cập vào một trang web sẽ được thực hiện như chính chúng ta sử dụng trình duyệt web hàng ngày. Đầu tiên, ta khởi tạo một driver sử dụng đoạn code sau:

```

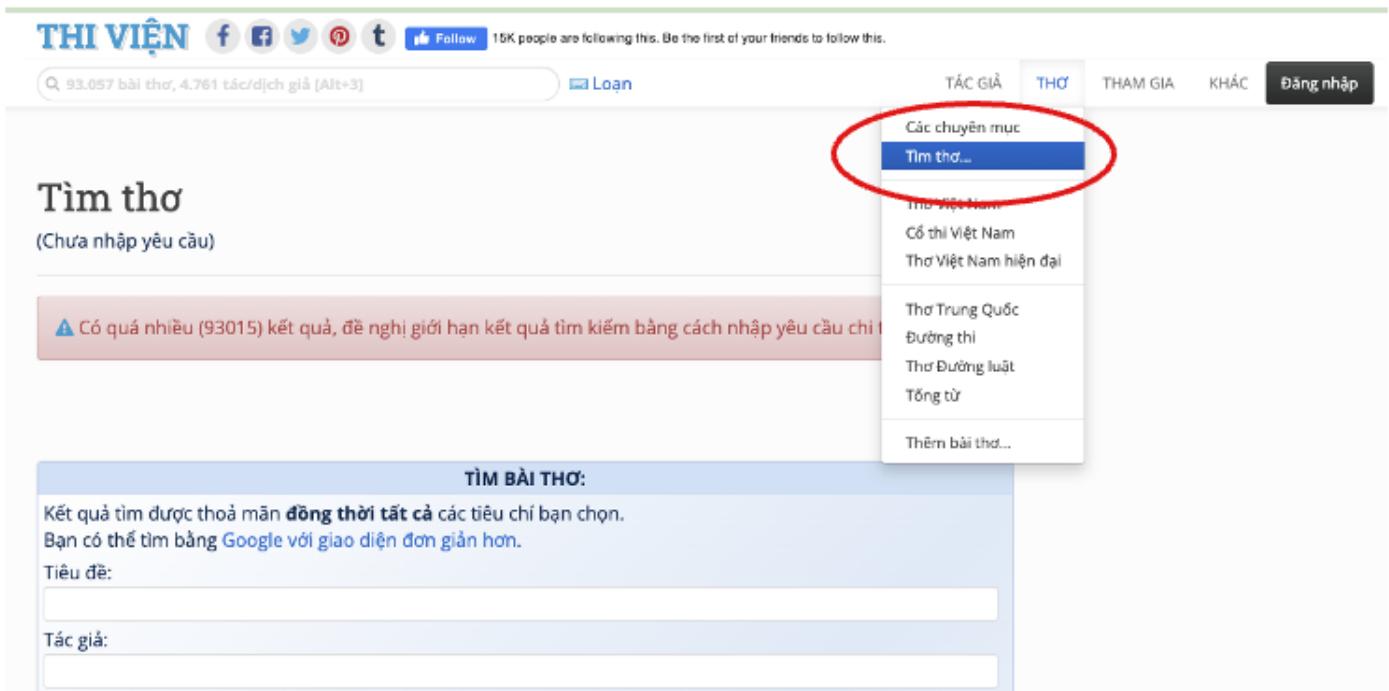
1 WEBDRIVER_DELAY_TIME_INT = 10
2 TIMEOUT_INT = 10
3 service = Service(executable_path=r"/usr/bin/chromedriver")
4 chrome_options = webdriver.ChromeOptions()
5 chrome_options.add_argument("--headless")
6 chrome_options.add_argument("--no-sandbox")
7 chrome_options.add_argument("--disable-dev-shm-usage")
8 chrome_options.add_argument("window-size=1920x1080")
9 chrome_options.headless = True
10 driver = webdriver.Chrome(service=service, options=chrome_options)
11 driver.implicitly_wait(TIMEOUT_INT)
12 wait = WebDriverWait(driver, WEBDRIVER_DELAY_TIME_INT)

```

Driver trong Selenium đóng vai trò như trình duyệt web, giúp ta thực hiện các thao tác như truy cập vào trang web dựa vào đường dẫn, thao tác chuyển trang...

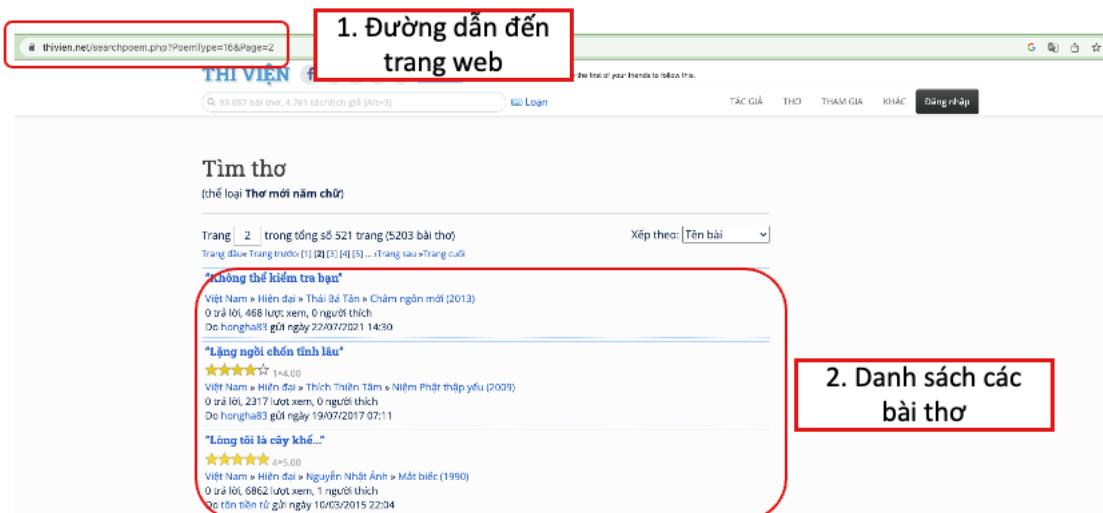
### II.1.4. Phân tích nội dung cần trích xuất

Để việc triển khai code được thuận lợi, ta cần xác định rõ kiến trúc file html của trang web cũng như các thành phần, nội dung mà ta mong muốn trích xuất. Một cách tìm nhanh chóng nhất đó là ta nên tìm đến trang tìm kiếm, từ đó sử dụng selenium để duyệt qua toàn bộ các bài viết được liệt kê trong trang tìm kiếm đó. Trong [thivien.net](#), ta chọn mục **Tìm thơ...** để đến trang này.



Hình 4: Minh họa vị trí tìm thơ.

Sau đó, các bạn hãy điền một số thông tin trong bảng **TÌM BÀI THƠ**; ở đây ta chỉ cần quan tâm đến trường thông tin **Thể thơ**: được chọn vào "**"Thơ mới năm chữ**". Sau khi bấm tìm kiếm, một trang web với các bài thơ với thể thơ ngũ ngôn xuất hiện.



Hình 5: Minh họa trang chứa danh sách các bài thơ năm chữ.

Tại đây, ta đã có được thông tin đường dẫn của trang web (mainpage\_url). Chúng ta sẽ sử dụng driver đã định nghĩa ở trên để truy cập vào trang này bằng lệnh **driver.get()**:

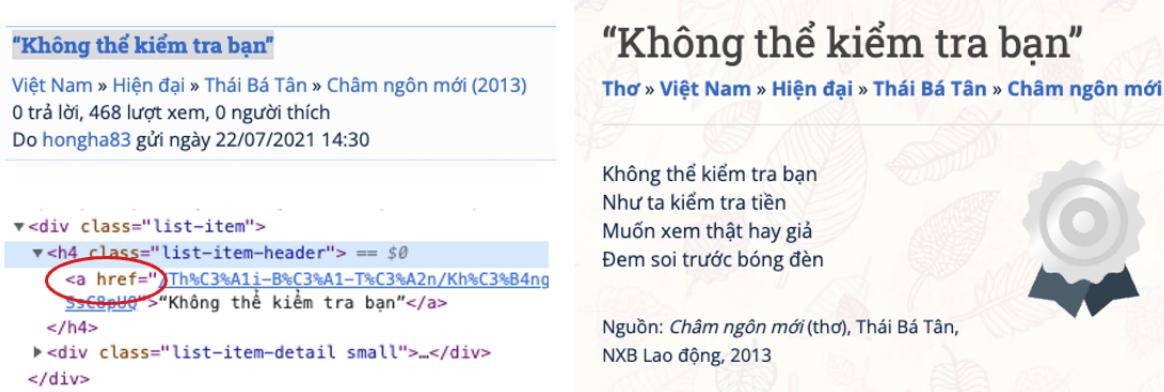
```

1 for page_idx in tqdm(range(1, 11)):
2     main_url = f "https://www.thivien.net/searchpoem.php?PoemType=16&ViewType=1
                  &Country=2&Age []=3&Page={page_idx}"
3     driver.get(main_url)

```

Nhận thấy đường dẫn trang web có chứa các trường thông tin để ta có thể di chuyển qua lại tại các trang tiếp theo của bảng tìm kiếm ('Page=2' tức đang ở trang 2 của bảng). Vì vậy ta có thể tận dụng điều này để tạo một vòng lặp lặp qua từng trang một cách tự động (với đoạn code trên ta sẽ duyệt từ trang thứ 1 đến trang thứ 10).

Ta tiếp tục phân tích các thành phần ta cần trích xuất đối với một trang thơ thông qua việc đọc cấu trúc html của trang web (cấu trúc html của trang web có thể được tìm thấy thông qua tính năng Inspect trên trình duyệt). Nhận thấy, thông tin duy nhất ta quan tâm đó chính là nội dung bài thơ. Song, để tiện cho các tác vụ sau này nếu có, ta sẽ trích xuất thêm các thông tin khác của bài thơ như Tựa đề, Nguồn...



The screenshot shows a search result for the poem "Không thể kiểm tra bạn". The result includes the title, author, date, and a snippet of the poem. Below the snippet, the HTML source code is shown with the href link to the poem highlighted and circled in red.

Hình 6: Ví dụ về một bài thơ được liệt kê trong bảng tìm kiếm và thẻ HTML của nó.

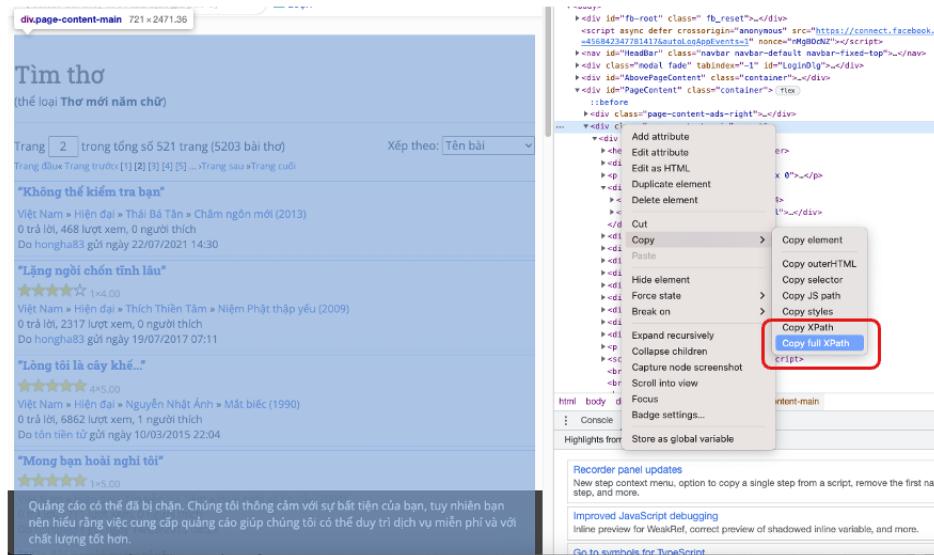
Như vậy, mỗi bài thơ được liệt kê tại trang tìm kiếm là một thẻ `<div>`, bên trong có chứa đường dẫn đến trang chứa bài thơ chính khi ta click chuột vào. Từ đây, ta dễ dàng truy cập vào mỗi bài thơ bằng cách đọc đường dẫn của thuộc tính 'href' chứa tại thẻ `<a>` (như hình). Để thực hiện được điều này, đầu tiên ta cần đọc được thẻ html chứa bảng dữ liệu, thực hiện như sau:

```

1 content_tags_xpath = '//*[@@class="page-content_container"]//div[@class="page-
content-main"]//div[@class="list-item"]
,
2 content_tags = driver.find_elements(By.XPATH, content_tags_xpath)

```

Trong Selenium, có nhiều cách để xác định và đọc thẻ html từ trang web thông qua hai phương thức `driver.find_element()` (tìm một thẻ khớp) và `driver.find_elements()` (tìm nhiều thẻ khớp) (chi tiết tại [đây](#)) song tìm kiếm bằng XPATH là một cách nhanh chóng nhất. Ở đây, ta quan tâm đến các thẻ div chứa thông tin bài thơ nên ta sẽ dùng `find_elements()` để tìm toàn bộ các thẻ này. Và với danh sách các thẻ của từng bài thơ (`content_tags`), ta đã có thể truy cập vào nội dung chi tiết của từng bài thơ.



Hình 7: Tìm XPATH của một thẻ HTML trên trình duyệt.

Khi gom hết tất cả các dữ kiện trên, ta có thể tạo một hàm dùng để bốc tách các trang web thơ với input là số thứ tự của trang danh sách thơ:

```

1 def extract_poem_links(driver, page_idx):
2     main_url = f"https://www.thivien.net/searchpoem.php?PoemType=16&ViewType=1
3                                     &Country=2&Age []=3&Page={page_idx}"
4
5     driver.get(main_url)
6     time.sleep(random.uniform(3, 5))
7
8     content_tags_xpath = '//*[@class="page-content container"]//div[@class="page-content-main"]//div[@class="list-item"]'
9
10    content_tags = driver.find_elements(By.XPATH, content_tags_xpath)
11    poem_links = []
12    for tag in content_tags:
13        try:
14            link_element = tag.find_element(By.XPATH, './h4[@class="list-item-header"]/a')
15            poem_title = link_element.text
16            poem_url = link_element.get_attribute("href")
17            poem_links.append({"title": poem_title, "url": poem_url})
18        except Exception as e:
19            print(f"Error extracting link: {e}")
20            continue
21
22    return poem_links

```

### II.1.5. Thực hiện trích xuất nội dung thơ

Cuối cùng, với từng thẻ html của trang thơ, ta sẽ thực hiện trích xuất các thông tin mà ta đã xác định (gồm Nội dung bài thơ, Tựa đề, Nguồn) của bài thơ tương ứng. Các thông tin này sẽ được lưu thành một dictionary và đẩy vào một list lưu trữ chung (datasets). Trước khi duyệt qua từng trang danh sách thơ, ta xây dựng sẵn một hàm chuyên dùng để tiền xử lý các nội dung

thơ sau khi thu thập được. Chức năng của hai hàm dưới đây chỉ dùng để loại bỏ một số nội dung không liên quan đến thơ như các thẻ HTML xen lẩn thơ:

```

1 def clean_poem_html(html):
2     html = re.sub(r"<img.*?", "", html, flags=re.IGNORECASE)
3     html = re.sub(r"<i>.*?</i>", "", html, flags=re.IGNORECASE | re.DOTALL)
4     html = re.sub(r"<b>(.*)</b>(?!>\s*(?:<br\s*/?>\s*){2,})", r"\1", html,
5                   flags=re.IGNORECASE)
6     html = re.sub(r"<br\s*/?>", "\n", html, flags=re.IGNORECASE)
7     html = re.sub(r"</?p>", "", html, flags=re.IGNORECASE)
8
9     return html.strip()
10
11
12
13 def process_poem_content(html, poem_src, poem_url, default_title=""):
14     cleaned = clean_poem_html(html)
15
16     pattern = re.compile(r"<b>(.*)</b>\s*\n{2,}", flags=re.IGNORECASE)
17     matches = list(pattern.finditer(cleaned))
18
19     poems = []
20     if matches:
21         for i, match in enumerate(matches):
22             title = match.group(1).strip()
23             start = match.end()
24             end = matches[i+1].start() if i + 1 < len(matches) else len(
25                                         cleaned)
26             content = cleaned[start:end].strip("\n")
27             poems.append({
28                 "title": title,
29                 "content": content,
30                 "source": poem_src,
31                 "url": poem_url
32             })
33     else:
34         poems.append({
35             "title": default_title,
36             "content": cleaned,
37             "source": poem_src,
38             "url": poem_url
39         })
40
41     return poems

```

Cuối cùng, ta định nghĩa hai hàm dùng để trích xuất nội dung của một đường dẫn chứa thơ năm chữ cũng như xây dựng hàm duyệt qua từng trang chứa thơ như sau:

```

1 def scrape_poem(driver, poem_url):
2     driver.get(poem_url)
3     time.sleep(random.uniform(3, 5))
4
5     poem_content_tag = WebDriverWait(driver, 10).until(
6         EC.visibility_of_element_located((By.CSS_SELECTOR, "div.poem-content"))
7     )
8
9     html_content = poem_content_tag.get_attribute("innerHTML")
10

```

```

11     try:
12         poem_src_tag = WebDriverWait(driver, 10).until(
13             EC.presence_of_element_located((By.XPATH, '//div[@class="small"]'))
14         )
15         poem_src = poem_src_tag.text
16     except Exception:
17         poem_src = ""
18
19     return process_poem_content(html_content, poem_src, poem_url)
20
21 def scrape_poems(driver, num_pages=10):
22     datasets = []
23     for page_idx in tqdm(range(1, num_pages + 1)):
24         poem_links = extract_poem_links(driver, page_idx)
25         for poem in poem_links:
26             poem_url = poem["url"]
27             try:
28                 poems = scrape_poem(driver, poem_url)
29                 datasets.extend(poems)
30             except Exception as e:
31                 print(f"Error processing {poem_url}: {e}")
32                 continue
33     return datasets

```

Khi mọi hàm đã sẵn sàng, việc còn lại của chúng ta là thực thi hàm `scrape_poems()` để bắt đầu quá trình thu thập thơ:

```

1 datasets = scrape_poems(driver, num_pages=10)
2 driver.quit()

```

Các kỹ thuật tại bước này đều xoay quanh việc tìm kiếm bằng XPATH kèm theo một số logic python phát sinh trong quá trình thử nghiệm code, các bạn nên đọc qua từng dòng cũng như kiểm nghiệm lại giá trị các biến dữ liệu để có thể hiểu tường tận các bước trích xuất trên. Thông qua đó, các bạn cũng có thể áp dụng để thu thập dữ liệu từ các trang web khác theo nhu cầu của các bạn (Ví dụ: thu thập các bài thơ lục bát tại [thivien.net...](http://thivien.net...))

### II.1.6. Lưu bộ dữ liệu thành file .csv

Sau khi hoàn tất giai đoạn thu thập trên, ta sẽ có một danh sách các dictionary (record) chứa thông tin của một bài thơ. Lúc này, để thuận tiện trong việc lưu trữ, ta sẽ lưu danh sách này thành một file .csv.

	title	content	source	url
0	"Bạn xấu như chiếc bóng"	Bạn xấu như chiếc bóng\nCứ bám riết theo anh\n...	[Thông tin 1 nguồn tham khảo đã được ấn]	<a href="https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1i...">https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1i...</a>
1	"Cái làm ta hạnh phúc"	Cái làm ta hạnh phúc\nThực ra cũng chẳng nhiều...	[Thông tin 1 nguồn tham khảo đã được ấn]	<a href="https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1i...">https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1i...</a>
2	"Chiều vừa xốp trên tay"	Chiều vừa xốp trên tay\nChợt nghe thoáng ong b...	[Thông tin 1 nguồn tham khảo đã được ấn]	<a href="https://www.thivien.net/L%C3%A2m-Huy-Nhu%...">https://www.thivien.net/L%C3%A2m-Huy-Nhu%...</a>
3	"Chơi thân không có nghĩa"	Chơi thân không có nghĩa\nKhông cãi nhau bao g...	[Thông tin 1 nguồn tham khảo đã được ấn]	<a href="https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1i...">https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1i...</a>
4	"Có thể buồn chút ít"	Có thể buồn chút ít\nMột mình, không người yêu...	[Thông tin 1 nguồn tham khảo đã được ấn]	<a href="https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1i...">https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1i...</a>
...	...	...	...	...

Hình 8: Dữ liệu thu thập được trong bảng dữ liệu.

```

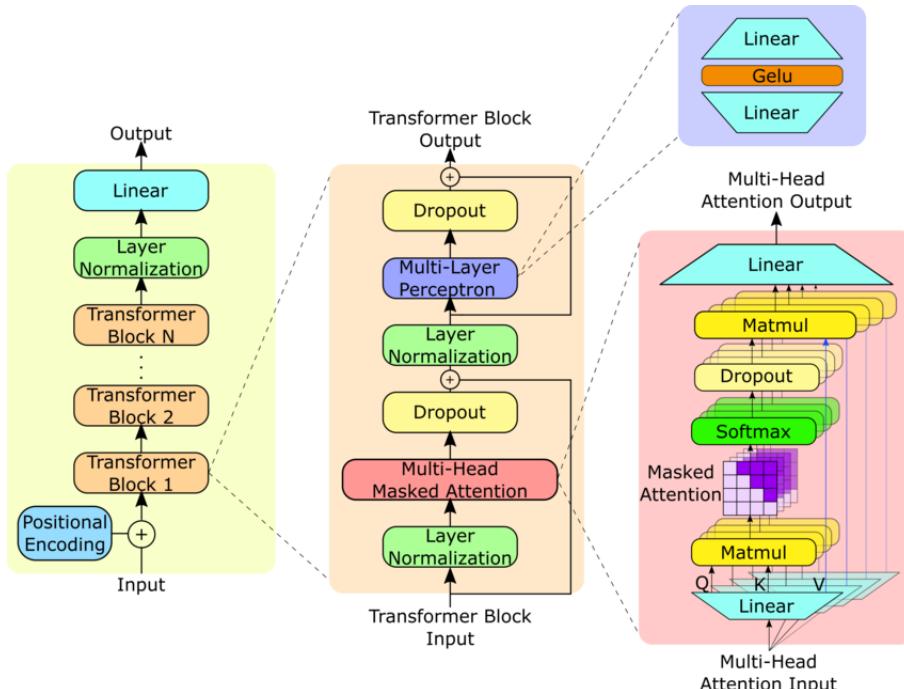
1 df = pd.DataFrame(datasets)
2 df.to_csv("poem_dataset.csv", index=True)

```

Trong phần code ví dụ trên, ta chỉ cài đặt để thu thập 10 trang đầu tiên trong trang tìm kiếm (vì trang web giới hạn số trang hiển thị).

## II.2. Xây dựng mô hình sinh thơ

Sau khi hoàn tất quá trình thu thập dữ liệu, chúng ta sẽ thực hiện huấn luyện (fine-tuning) mô hình GPT-2. GPT-2 (Generative Pre-trained Transformer 2) là một mô hình ngôn ngữ lớn phiên bản thứ 2 trong chuỗi các mô hình họ GPT được phát triển bởi OpenAI. GPT-2 được xây dựng dựa trên kiến trúc Transformer Decoder-only, các bạn có thể coi ảnh minh họa kiến trúc của GPT-2 ở hình dưới đây:

Hình 9: Kiến trúc mô hình của GPT-2. Nguồn: [link](#).

Ở phần này, chúng ta sẽ sử dụng mô hình GPT2 để thực hiện fine-tuning cho mục đích sinh thơ tiếng Việt. Theo đó, các bước thực hiện như sau:

### II.2.1. Tải các thư viện cần thiết

```
1 !pip install -qq datasets evaluate transformers[sentencepiece]
2 !pip install -qq accelerate
3 !apt install git-lfs
```

### II.2.2. Import các thư viện cần thiết

Chúng ta sẽ sử dụng thư viện HuggingFace với 2 module quan trọng là GPT2Tokenizer và GPT2LMHeadModel.

```
1 import os
2 import math
3 import torch
4 import pandas as pd
5
6 from transformers import GPT2Tokenizer, GPT2LMHeadModel
7 from transformers import DataCollatorForLanguageModeling
8 from transformers import TrainingArguments, Trainer
9 from huggingface_hub import notebook_login
10 from datasets import Dataset
```

### II.2.3. Load bộ dữ liệu

Với bộ dữ liệu đã thu thập được, chúng ta sẽ tiến hành đọc file .csv lên như sau:

```
1 DATASET_PATH = "poem_final.csv"
2 df = pd.read_csv(DATASET_PATH)
3 df
```

### II.2.4. Chuẩn bị bộ dữ liệu

Hiện tại, đoạn thơ ta tách được có cấu trúc như sau:

Áo chiều nào còn vương  
 Người ấy đến rất nhớ  
 Từ đất đen đứng dậy  
 Vào cuộc đời đáng yêu

Một khổ thơ

Người đi trong nắng mới  
 Gió hát khúc yêu đời  
 Mùa xuân xanh lá thắm  
 Hy vọng mãi không rời.

Hình 10: Cấu trúc của một mẫu dữ liệu (một bài thơ năm chữ).

Một bài thơ có thể có nhiều khổ thơ (part), một khổ thơ sẽ gồm 4 dòng thơ, mỗi dòng gồm 5 chữ. Để giảm độ phức tạp của bài toán, chúng ta sẽ coi mỗi khổ thơ là một data sample, và dùng chúng cho việc huấn luyện mô hình. Đầu tiên, ta xây dựng hàm tách nội dung thơ của một mẫu dữ liệu thành các danh sách chứa 4 dòng thơ:

```

1 def split_content(content):
2     samples = []
3
4     poem_parts = content.split("\n\n")
5     for poem_part in poem_parts:
6         poem_in_lines = poem_part.split("\n")
7         if len(poem_in_lines) == 4:
8             samples.append(poem_in_lines)
9
10    return samples
11
12 df["content"] = df["content"].apply(lambda x: split_content(x))
13 df

```

Nhận thấy nội dung cột content của mỗi hàng dữ liệu là một list chứa các sublist. Ta sẽ thực hiện tách các sublist này thành một hàng trong bảng dữ liệu mới, coi như là một sample mới trong bộ dữ liệu. Cách thực hiện như sau:

```

1 df_exploded = df.explode("content")
2 df_exploded.reset_index(drop=True, inplace=True)
3 df_exploded = df_exploded.dropna(subset=["content"])
4 df_exploded

```

Hàm `df.explode()` sẽ giúp ta tách các phần tử trong một list thành các hàng mới. Khi thực thi xong đoạn code trên, ta có bảng dữ liệu mới như sau:

Unnamed: 0		title	content	source	url
0	0	"Cái làm ta hạnh phúc"	[Cái làm ta hạnh phúc, Thực ra cũng chẳng nhiều...	Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB L...	<a href="https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3...">https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3...</a>
1	0	"Cái làm ta hạnh phúc"	[Rồi thêm chút công việc, Cho ta làm hàng ngày...	Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB L...	<a href="https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3...">https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3...</a>
2	1	"Chiều vừa xốp trên tay"	[Chiều vừa xốp trên tay, Chợt nghe thoảng ong ...	Nguồn: Lâm Huy Nhuận, Chiều có thật (thơ), NXB...	<a href="https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...">https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...</a>
3	1	"Chiều vừa xốp trên tay"	[Ớt đỏ sao cứ đỏ, Táo chín cho thật vàng, Em đ...	Nguồn: Lâm Huy Nhuận, Chiều có thật (thơ), NXB...	<a href="https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...">https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...</a>
4	2	"Dưới giàn hoa thiên lý..."	[Dưới giàn hoa thiên lý, Một mình anh đang ngồi...	Nguồn: Nguyễn Nhật Ánh, Mắt biếc, NXB Trẻ, 2004	<a href="https://www.thivien.net/Nguy%E1%BB%85n-Nh%E1%BA...">https://www.thivien.net/Nguy%E1%BB%85n-Nh%E1%BA...</a>
...	...	...	...	...	...

Hình 11: Bảng dữ liệu sau khi sử dụng hàm explode của pandas để tách các khổ thơ thành các hàng dữ liệu mới. Có thể nhận thấy số hàng trong bảng dữ liệu đã tăng lên.

Ta cần nội dung thơ (giá trị của cột content) phải ở dạng string. Vì vậy, ta sẽ thực hiện convert nội dung content sang string như sau:

```
1 df_exploded["content"] = df_exploded["content"].apply(lambda x: "\n".join(x))
2 df_exploded
```

Unnamed: 0		title	content	source	url
0	0	"Cái làm ta hạnh phúc"	Cái làm ta hạnh phúc\nThực ra cũng chẳng nhiều...	Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB L...	<a href="https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3...">https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3...</a>
1	0	"Cái làm ta hạnh phúc"	Rồi thêm chút công việc\nCho ta làm hàng ngày...	Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB L...	<a href="https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3...">https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3...</a>
2	1	"Chiều vừa xốp trên tay"	Chiều vừa xốp trên tay\nChợt nghe thoảng ong b...	Nguồn: Lâm Huy Nhuận, Chiều có thật (thơ), NXB...	<a href="https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...">https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...</a>
3	1	"Chiều vừa xốp trên tay"	Ớt đỏ sao cứ đỏ\nTáo chín cho thật vàng\nEm đ...	Nguồn: Lâm Huy Nhuận, Chiều có thật (thơ), NXB...	<a href="https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...">https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...</a>
4	2	"Dưới giàn hoa thiên lý..."	Dưới giàn hoa thiên lý\nMột mình anh đang ngồi...	Nguồn: Nguyễn Nhật Ánh, Mắt biếc, NXB Trẻ, 2004	<a href="https://www.thivien.net/Nguy%E1%BB%85n-Nh%E1%BA...">https://www.thivien.net/Nguy%E1%BB%85n-Nh%E1%BA...</a>
...	...	...	...	...	...

Hình 12: Bảng dữ liệu sau khi chuyển đổi nội dung cột content sang dạng string.

Với DataFrame đã chuẩn bị xong, chúng ta sẽ đổi dạng dữ liệu pandas này sang HuggingFace dataset để thuận tiện trong việc sử dụng thư viện:

```
1 TEST_SIZE = 0.1
2 poem_dataset = Dataset.from_pandas(df_exploded)
3 poem_dataset = poem_dataset.train_test_split(test_size=TEST_SIZE)
```

```

1 poem_dataset = Dataset.from_pandas(df_exploded)
2 poem_dataset

Dataset({
    features: ['Unnamed: 0', 'title', 'content', 'source', 'url', '__index_level_0__'],
    num_rows: 441
})

1 TEST_SIZE = 0.1
2 poem_dataset = poem_dataset.train_test_split(test_size=TEST_SIZE)
3 poem_dataset

DatasetDict({
    train: Dataset({
        features: ['Unnamed: 0', 'title', 'content', 'source', 'url', '__index_level_0__'],
        num_rows: 396
    })
    test: Dataset({
        features: ['Unnamed: 0', 'title', 'content', 'source', 'url', '__index_level_0__'],
        num_rows: 45
    })
})

```

Hình 13: Bảng dữ liệu sau khi được đổi sang HuggingFace dataset.

### II.2.5. Tiền xử lý dữ liệu

Với bộ dữ liệu thơ đã sẵn sàng, chúng ta bắt đầu quy trình tiền xử lý bộ dữ liệu để chuẩn bị cho việc huấn luyện mô hình. Đầu tiên, ta khai báo tokenizer:

```

1 MODEL_NAME = "danghuy1999/gpt2-viwiki"
2
3 tokenizer = GPT2Tokenizer.from_pretrained(MODEL_NAME)

```

Sau đó, xây dựng hàm chạy tokenization cho mỗi sample và thực thi chúng lên bộ dữ liệu:

```

1 tokenizer.pad_token = tokenizer.eos_token
2 MAX_SEQ_LEN = 100
3
4 def preprocess_function(row):
5     return tokenizer(
6         row["content"],
7         max_length=MAX_SEQ_LEN,
8         padding="max_length",
9         truncation=True
10    )
11
12 tokenized_poem_dataset = poem_dataset.map(
13     preprocess_function,
14     batched=True,
15     num_proc=4,
16     remove_columns=poem_dataset["train"].column_names,
17 )

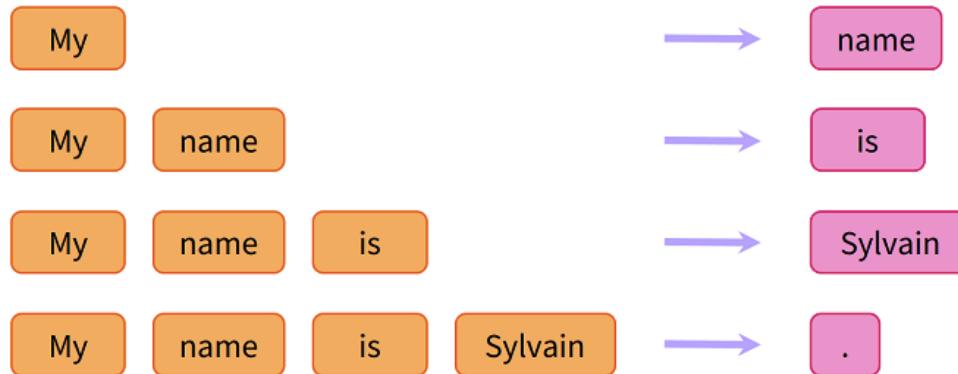
```

Khi huấn luyện mô hình ngôn ngữ trong HuggingFace, chúng ta sẽ khai báo một instance từ class DataCollatorForLanguageModeling. Việc này nhằm giúp HuggingFace hỗ trợ chúng ta việc batching dữ liệu để việc huấn luyện mô hình ngôn ngữ trở nên hiệu quả hơn.

```

1 data_collator = DataCollatorForLanguageModeling(tokenizer=tokenizer,
2                                               mlm=False)

```



Hình 14: Ảnh minh họa việc đoán từ tiếp theo dựa trên các từ trước đó. Nguồn: [link](#).

### II.2.6. Huấn luyện mô hình

Với bộ dữ liệu đã được tiền xử lý, ta sẽ bắt đầu việc huấn luyện mô hình. Đầu tiên, ta load pre-trained model GPT-2:

```

1 model = GPT2LMHeadModel.from_pretrained(MODEL_NAME)

```

Sau đó, khai báo một vài config trong việc huấn luyện mô hình:

```

1 training_args = TrainingArguments(
2     output_dir="gpt2_viet_poem_generation",
3     save_strategy="epoch",
4     learning_rate=2e-5,
5     num_train_epochs=10,
6     weight_decay=0.01,
7     fp16=True
8 )

```

Cuối cùng, ta thực thi trainer để tiến hành training:

```

1 trainer = Trainer(
2     model=model,
3     args=training_args,
4     train_dataset=tokenized_poem_dataset["train"],
5     eval_dataset=tokenized_poem_dataset["test"],
6     data_collator=data_collator,
7     tokenizer=tokenizer
8 )
9
10 trainer.train()

```

Như vậy, sau khi hoàn thành các bước trên, ta đã hoàn tất quá trình huấn luyện một mô hình sinh thơ tiếng Việt.

## II.2.7. Inference

Để sử dụng mô hình này, chúng ta có thể đưa lên HuggingFace hub và gọi mô hình xuống để sử dụng hoặc chúng ta có thể làm theo cách sau:

```

1 prompt = "Học học nữa học mãi\n"
2 device = "cuda" if torch.cuda.is_available() else "cpu"
3 inputs = tokenizer(prompt, return_tensors="pt").input_ids.to(device)
4 outputs = model.generate(
5     inputs,
6     max_new_tokens=50,
7     do_sample=True,
8     top_k=50,
9     top_p=0.95,
10    temperature=0.8,
11    repetition_penalty=1.2
12)
13 results = tokenizer.batch_decode(outputs, skip_special_tokens=True)
14 results = results[0]
15 print()
16 for line in results.split("\n"):
17     print(line)

```

Demo về một đoạn thơ năm chữ được tạo sinh từ mô hình GPT-2 sau khi thực hiện fine-tuning của chúng ta:

*Học học nữa học mãi  
Hàng trăm ngàn đêm dài  
Nhưng, không ai quên nghỉ  
Quanh ta thấy chuyện gì?*

— Fine-tuned GPT-2 on a tiny Vietnamese Five-Word Poetry Dataset

Với mô hình xây dựng được, ta có thể triển khai thành một ứng dụng web demo chương trình cho phép tạo sinh một bài thơ bắt đầu từ một vài từ cho trước. Web demo còn được cài đặt để ta có thể điều chỉnh các thông số kỹ thuật có thể làm thay đổi kết quả tạo sinh của mô hình GPT-2, giúp người dùng có thể thấy được đa dạng các kết quả hơn từ mô hình. Thông tin về web demo có thể được tìm thấy tại phần IV.



# Vietnamese Poem Generation

The used model is GPT-2 trained on Vietnamese poems: thangduong0509/gpt2\_viet\_poem\_generation

Instructions ▼

Enter a few words or sentences to start:

Con song que toi dep

Max Output Tokens:  75

Temperature:  0.80

Top-k:  50

Generate Poem

**Generated Poem:**

Con song que toi dep

Tiếng mèo người đâu xưa

nguyễn hoa biển lạnh?

Đôi mắt dài vền!

Hình 15: Ảnh minh họa web demo cho chương trình tạo sinh thơ tiếng Việt.

### III. Câu hỏi trắc nghiệm

1. Mô hình Text Generation là?
  - (a) Mô hình sinh chữ từ ảnh.
  - (b) Mô hình sinh chữ từ video.
  - (c) Mô hình sinh chữ từ một input nào đó.
  - (d) Mô hình sinh chữ từ bản ghi âm thanh.
2. Ứng dụng nào sau đây thuộc về Text Generation?
  - (a) Image Captioning.
  - (b) Text Summarization.
  - (c) Automatic Speech Recognition.
  - (d) Tất cả các phương án trên.
3. Mục tiêu của bài toán Text Generation là?
  - (a) Copy văn bản đầu vào.
  - (b) Tạo văn bản mới dựa trên dữ liệu đầu vào.
  - (c) Sửa văn bản đầu vào.
  - (d) Không phương án nào đúng.
4. Các thách thức trong bài Text Generation?
  - (a) Văn bản đầu ra có ngữ pháp đúng.
  - (b) Văn bản đầu ra có nghĩa.
  - (c) Văn bản đầu ra phải mạch lạc.
  - (d) Tất cả các phương án trên.
5. Khi mô hình được thiết kế để dự đoán từ tiếp theo dựa trên một chuỗi các từ trước đó, ta gọi bài toán này là gì?
  - (a) Causal Language Modeling.
  - (b) Masked Language Modeling.
  - (c) Sequence to Sequence.
  - (d) Denoising.
6. Mô hình GPT2 được xây dựng dựa theo kiến trúc nào?
  - (a) Transformer Encoder-Decoder.
  - (b) Transformer Encoder-only.
  - (c) Transformer Decoder-only.
  - (d) Long Short-Term Memory.

7. Selenium là?
- (a) Ngôn ngữ lập trình.
  - (b) Trình duyệt web.
  - (c) Một thư viện trong Python.
  - (d) Thiết kế mô hình học sâu.
8. Selenium thường được sử dụng trong việc?
- (a) Thu thập dữ liệu trên web.
  - (b) Thiết kế giao diện web.
  - (c) Tối ưu siêu tham số mô hình học sâu.
  - (d) Thiết kế mô hình học sâu.
9. Dòng lệnh nào sau đây dùng để truy cập vào một trang web trong Selenium với đường dẫn cho trước:
- (a) `driver.get()`
  - (b) `driver.switch_to_window()`
  - (c) `driver.execute_script()`
  - (d) `driver.close()`
10. Dòng lệnh nào sau đây dùng để tìm một thẻ HTML trong Selenium:
- (a) `driver.get()`
  - (b) `driver.find_element()`
  - (c) `driver.back()`
  - (d) `driver.execute_async_script()`

## IV. Phụ lục

1. **Datasets:** Các file dataset được đề cập trong bài có thể được tải tại [đây](#).
2. **Hint:** Các file code gợi ý có thể được tải tại [đây](#).
3. **Solution:** Các file code cài đặt hoàn chỉnh và phần trả lời nội dung trắc nghiệm có thể được tải tại [đây](#) (**Lưu ý:** Sáng thứ 3 khi hết deadline phần project, ad mới copy các nội dung bài giải nêu trên vào đường dẫn).
4. **Demo:** Web demo và mã nguồn của ứng dụng có thể được truy cập tại [đây](#).

**5. Rubric:**

Mục	Kiến Thức	Đánh Giá
II.	<ul style="list-style-type: none"> <li>- Kiến thức về crawl data.</li> <li>- Kiến thức về bài toán Text Generation.</li> <li>- Kiến thức về thư viện Selenium để crawl dữ liệu và thư viện HuggingFace để fine-tune mô hình ngôn ngữ.</li> </ul>	<ul style="list-style-type: none"> <li>- Nắm được cách sử dụng thư viện Selenium để lấy dữ liệu từ một trang web.</li> <li>- Nắm được cách sử dụng thư viện HuggingFace để fine-tune mô hình ngôn ngữ cho bài toán sinh thơ.</li> </ul>
III.	<ul style="list-style-type: none"> <li>- Các kiến thức cơ bản về bài toán Text Generation.</li> <li>- Các kiến thức cơ bản về thư viện Selenium trong Python.</li> <li>- Một số phương thức, hàm cơ bản trong Selenium.</li> </ul>	<ul style="list-style-type: none"> <li>- Hiểu được các khái niệm cơ bản về Selenium.</li> <li>- Cách sử dụng một số chức năng cơ bản về Selenium.</li> <li>- Hiểu được các khái niệm cơ bản của bài toán Text Generation.</li> </ul>

**- *Hết* -**